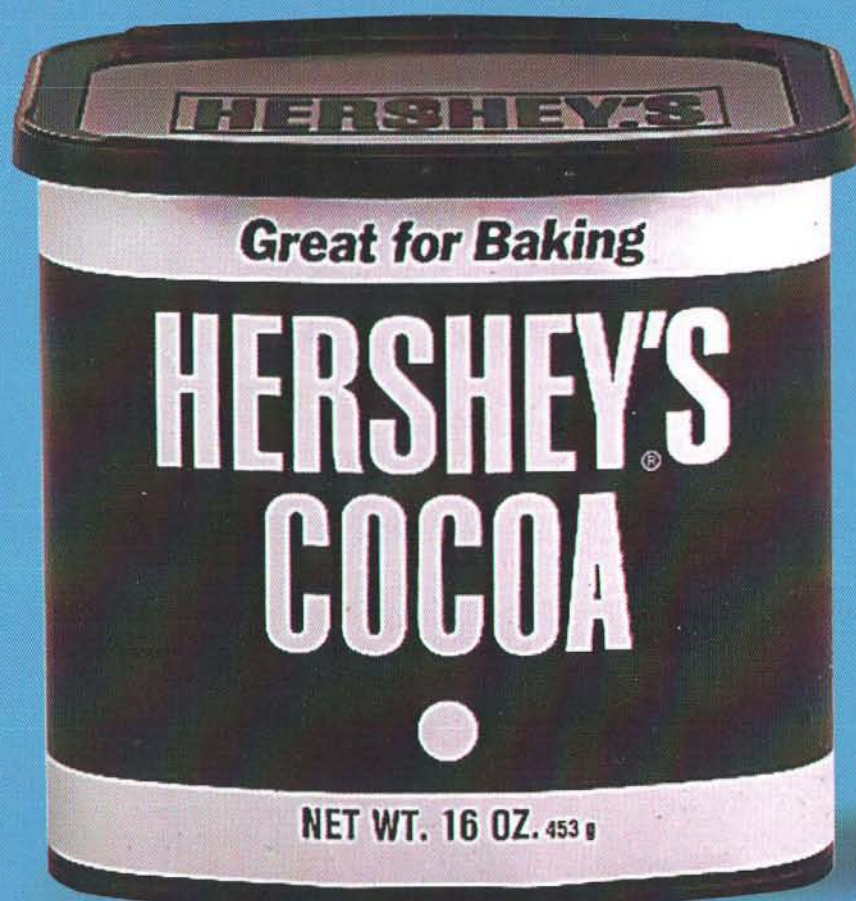


MacTech®

The Journal of Macintosh Technology and Development

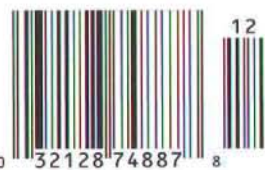
COCOA for JAVA DEVELOPERS



Hershey's is a registered trademark used with the permission of Hershey Foods Corporation.

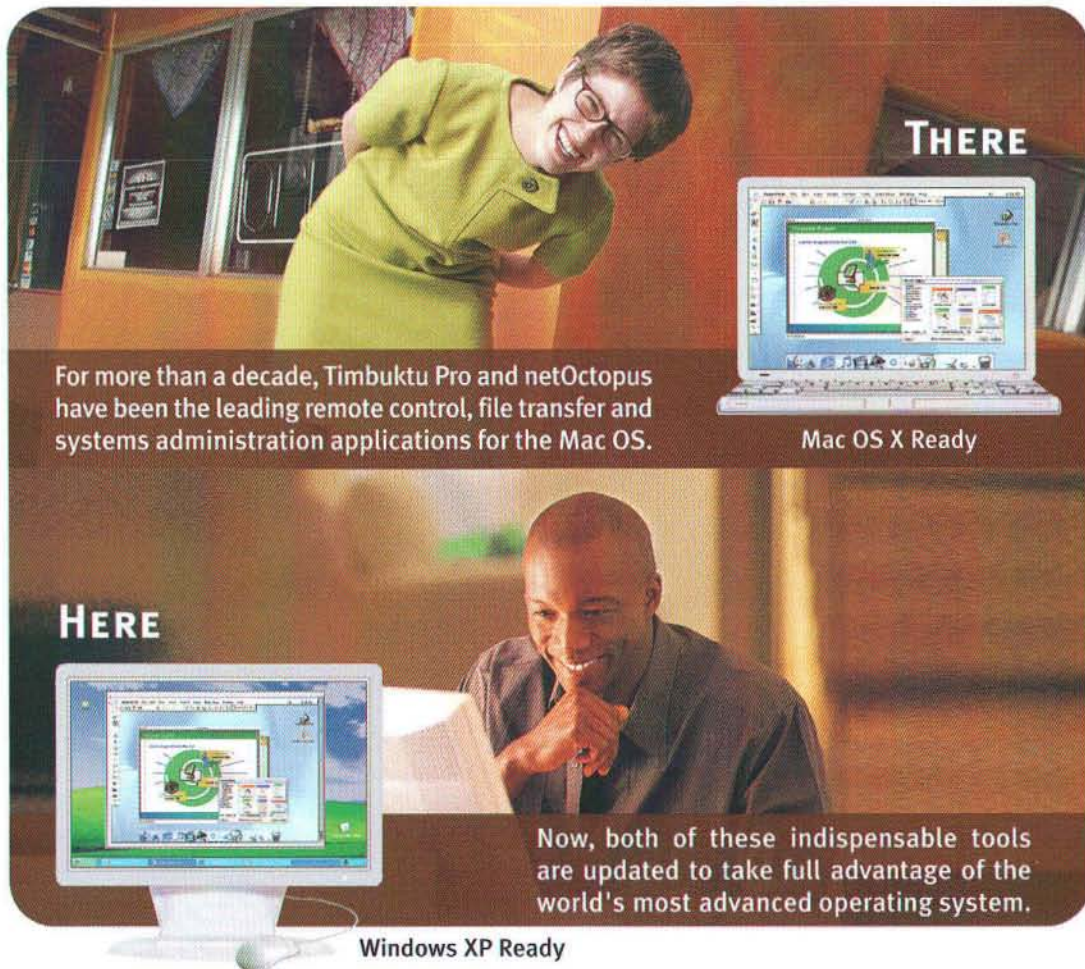
Writing Cocoa Applications in Java

by Steve Klingsporn



\$8.95 US
\$12.95 Canada
ISSN 1067-8360
Printed in U.S.A.

Stay In Control Wherever You Go.



THERE

For more than a decade, Timbuktu Pro and netOctopus have been the leading remote control, file transfer and systems administration applications for the Mac OS.

Mac OS X Ready

HERE

Now, both of these indispensable tools are updated to take full advantage of the world's most advanced operating system.

Windows XP Ready

Timbuktu Pro

Whether you're at home or at work, Timbuktu Pro allows you to operate distant computers as if you were sitting in front of them, transfer files or folders quickly and easily, and communicate by instant message, text chat, or voice intercom.

<http://www.timbuktopro.com>

netOctopus

Intuitive and powerful, netOctopus can manage a network of ten or 10,000 computers. Inventory computers, software and devices on your network; distribute software; configure remote computers; and create custom reports on the fly.

<http://www.netoctopus.com>

Learn more, try it, or buy it online. Call us at **1-800-485-5741**.



timbuktu® • netOctopus®

netopia®

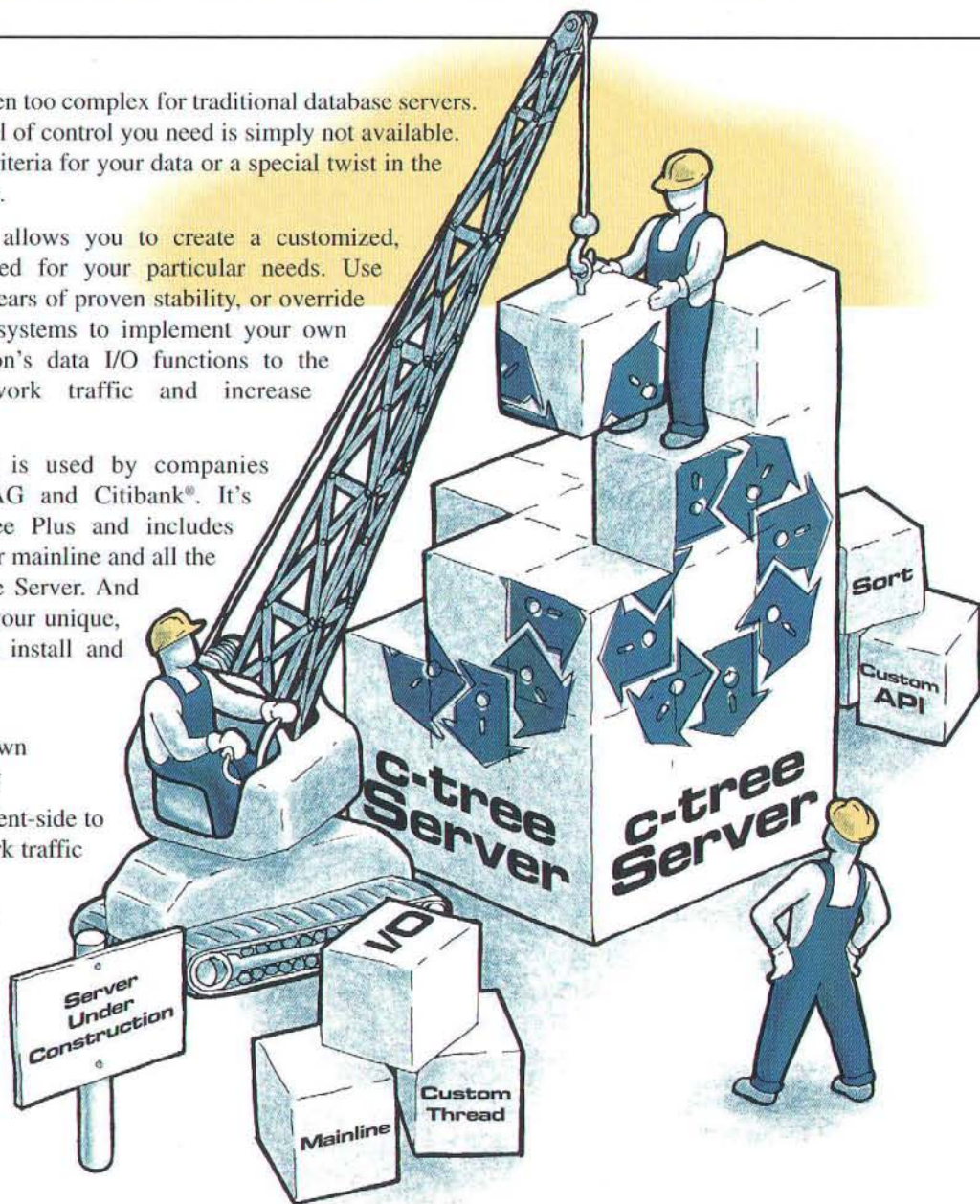
CUSTOMIZE YOUR DATABASE SERVER WITH THE C-TREE® SERVER SDK

Today's database demands are often too complex for traditional database servers. The functionality and precise level of control you need is simply not available. Perhaps you need alternate sort criteria for your data or a special twist in the threading or communication logic.

FairCom's c-tree® Server SDK allows you to create a customized, industrial-strength server designed for your particular needs. Use FairCom's kernel, with over 20 years of proven stability, or override functionality within specific subsystems to implement your own subtleties. Move your application's data I/O functions to the server-side to decrease network traffic and increase performance!

FairCom's c-tree Server SDK is used by companies worldwide such as Software AG and Citibank®. It's integrated seamlessly into c-tree Plus and includes complete source code to the server mainline and all the interface subsystems to the c-tree Server. And best of all, once you've created your unique, customized server, it is easy to install and administer: no DBA required!

- Enhance our server with your own custom server-side functionality
- Move functionality from the client-side to the server-side to reduce network traffic and increase performance
- Modify or replace entire server subsystems
- Complete source for the server mainline, key server subsystems, and client-side
- Flexible OEM licensing



Visit www.faircom.com/ep/mt/sdk today to take control of your server!



FairCom®
www.faircom.com

USA	573.445.6833
EUROPE	+39.035.773.464
JAPAN	+81.59.229.7504
BRAZIL	+55.11.3872.9802

DBMS Since 1979 • 800.234.8180 • info@faircom.com

Other company and product names are registered trademarks or trademarks of their respective owners.

© 2002 FairCom Corporation



Adaptive Server
Enterprise Now
Available on
Mac OS X 10.3
Panther!

HELPING FILEMAKER PRO CUSTOMERS SCALE TO NEW HEIGHTS.

Want to enhance the performance of FileMaker Pro on MAC OS X? The same database engine that runs Wall Street can help you do just that.

ASE 12.5 increases the reliability, availability and scalability of your FileMaker Pro application. It provides better data integrity, world-class security and the ability to handle thousands of transactions per minute. It'll also give your users the power of SQL queries.

ASE 12.5 for MAC OS X is yet one more example of how

Sybase is helping today's leading companies achieve Information Liquidity: a highly profitable state where all of your information is transformed into real economic value.

A FREE Developer's Edition download is available online at sybase.com/filemaker.

Free Developer Edition available for a limited time only. ©2003 Sybase, Inc. All rights reserved. All trademarks are the property of their respective owners.

INFORMATION LIQUIDITY.



SYBASE

BETTER WHEN EVERYTHING WORKS TOGETHER.™

How To Communicate With Us

In this electronic age, the art of communication has become both easier and more complicated. Is it any surprise that we prefer **e-mail**?

If you have any questions, feel free to call us at 805/494-9797 or fax us at 805/494-9798.

If you would like a subscription or need customer service, feel free to contact Developer Depot Customer Service at 877-MACTECH

DEPARTMENTS

Orders, Circulation, & Customer Service

Press Releases

Ad Sales

Editorial

Programmer's Challenge

Online Support

Accounting

Marketing

General

Web Site (articles, info, URLs and more...)

E-Mail/URL

cust_service@mactech.com

press_releases@mactech.com

ad_sales@mactech.com

editorial@mactech.com

prog_challenge@mactech.com

online@mactech.com

accounting@mactech.com

marketing@mactech.com

info@mactech.com

http://www.mactech.com

The MacTech Editorial Staff

Publisher • Neil Ticktin

Editor-in-Chief • Dave Mark

Managing Editor • Jessica Stubblefield

Regular Columnists

Getting Started

by Dave Mark

QuickTime Toolkit

by Tim Monroe

Mac OS X Programming Secrets

by Scott Knaster

Reviews/KoolTools

by Michael R. Harvey

Patch Panel

by John C. Welch

Section 7

by Rich Morin

Untangling the Web

by Kevin Hemenway

John and Pals' Puzzle Page

by John Vink

Regular Contributors

Vicki Brown, Erick Tejkowski,
Paul E. Seving

MacTech's Board of Advisors

Jordan Dea-Mattson, Jim Straus
and Jon Wiederspan

MacTech's Contributing Editors

- Michael Brian Bentley
- Vicki Brown
- Marshall Clow
- John C. Daub
- Tom Djajadiningrat
- Bill Doernfeld, Blueworld
- Andrew S. Downs
- Richard V. Ford, Packeteer
- Gordon Garb, Sun
- Ilene Hoffman
- Chris Kilbourn, Digital Forest
- Rich Morin
- John O'Fallon, Maxum Development
- Will Porter
- Avi Rappoport, Search Tools Consulting
- Ty Shipman, Kagi
- Chuck Shotton, BIAP Systems
- Cal Simone, Main Event Software
- Steve Sisak, Codewell Corporation
- Chuck Von Rospach, Plaidworks

Xplain Corporation Staff

Chief Executive Officer • Neil Ticktin

President • Andrea J. Sniderman

Controller • Michael Friedman

Production Manager • Jessica Stubblefield

Production/Layout • Darryl Smart

Marketing Manager • Nick DeMello

Account Executive • Lorin Rivers
adsales@mactech.com • 800-5-MACDEV

Events Manager • Susan M. Worley

International • Rose Kemps

Network Administrator • David Breffitt

Accounting • Jan Webber, Marcie Moriarty

Customer Relations • Laura Lane, Susan Pomrantz

Shipping/Receiving • Joel Licardie

Board of Advisors • Steven Geller, Blake Park,
and Alan Carsrud

All contents are Copyright 1984-2003 by Xplain Corporation. All rights reserved. MacTech and Developer Depot are registered trademarks of Xplain Corporation. RadGad, Useful Gifts and Gadgets, Xplain, DevDepot, Depot, The Depot, Depot Store, Video Depot, Movie Depot, Palm Depot, Game Depot, Flashlight Depot, Explain It, MacDev-1, THINK Reference, NetProfessional, NetProLive, JavaTech, WebTech, BeTech, LinuxTech, MacTech Central and the MacTutorMan are trademarks or service marks of Xplain Corporation. Sprocket is a registered trademark of eSprocket Corporation. Other trademarks and copyrights appearing in this printing or software remain the property of their respective holders.

MacTech Magazine (ISSN: 1067-8360 / USPS: 010-227) is published monthly by Xplain Corporation, 850-P Hampshire Road, Westlake Village, CA 91361-2800. Voice: 805/494-9797, FAX: 805/494-9798. Domestic subscription rates are \$47.00 per year. Canadian subscriptions are \$59.00 per year. All other international subscriptions are \$97.00 per year. Domestic source code disk subscriptions are \$77 per year. All international disk subscriptions are \$97.00 a year. Please remit in U.S. funds only. Periodical postage is paid at Thousand Oaks, CA and at additional mailing office.

POSTMASTER: Send address changes to **MacTech Magazine**, P.O. Box 5200, Westlake Village, CA 91359-5200.

C o n t e n t s

December 2003 • Volume 19, Issue 12

QUICKTIME TOOLKIT

44 The Matrix Revolutions

Handling Movie File Operations with Revolution

By Tim Monroe

COVER STORY

32 Writing Cocoa Applications in Java

By Steve Klingsporn

STARTING A BUSINESS

75 Pursuing the Dream

From Dream to Reality

By Chris Kilbourn

CASTING YOUR .NET

16 Compiling, disassembling and re-assembling .NET binaries

Exploring .NET development on Mac OS X

By Andrew Troelsen, Minneapolis, Minnesota

SOFTWARE MARKETING

70 Driving Traffic to Your Web Site

Online Publicity for Your Software

By Dave Wooldridge

PATCH PANEL

66 New Networking Tricks in Panther

A quick look at some new tricks, good and bad, in Panther

By John C. Welch

GETTING STARTED

6 Meet Andrew Welch

By Dave Mark, Editor In Chief

JAVA

62 RemoteScriptRunner: Remotely execute scripts from any web browser

By Joe Zobkiw

COCOA

26 A simple debugging tool for Cocoa

Another weapon in the fight against bugs

By Steve Gehrman

SECTION 7

42 ESR, etc.

A multiplex review of a singular individual...

By Rich Morin

MAC OS X PROGRAMMING SECRETS

12 Checking Out Kiosk Mode Features

By Scott Knaster

UNTANGLING THE WEB

56 Panther Roars, Database Meows

Panther worth your time? Databases besides Filemaker? Pffftttt.

By Kevin Hemenway, Titillating Entertainer

Copyright 2003 by Dave Mark

Meet Andrew Welch

This month's column is a bit of a departure from our normal Mac dev exploration. As long-time readers know, I am a *big* gaming fan. I recently had the chance to hook up with one of the originators of shareware gaming on the Mac and I wanted to share this conversation with you.

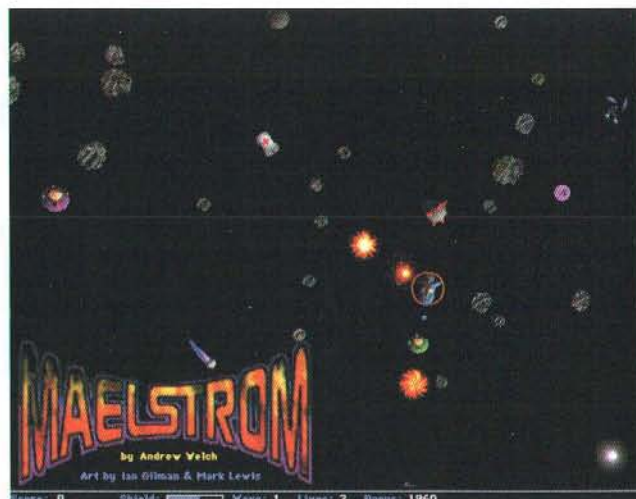
Andrew Welch, aka Ambrosia Software, Inc's el Presidente, started developing software in high school because frankly, it was better than flipping burgers (though he did have a stint as drive-thru guy at Roy Rogers). He went on to study Photojournalism in college, but decided to pursue software development for a living rather than carry someone else's camera bags. He currently lives in Rochester, NY, and has the oddly polar hobbies of origami and weight lifting.

Dave: Can I call you Andrew, or would you prefer el Presidente?

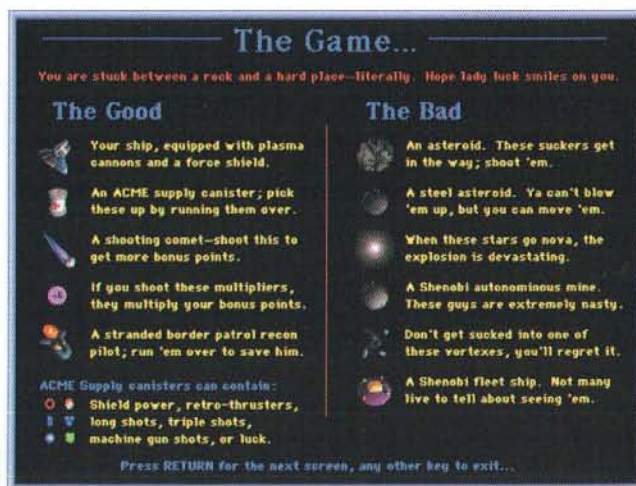
Andrew: If you call me "el Presidente" I might end up the target of some sort of "regime change", so let's keep it safe; "Andrew" is fine.

Dave: I first became aware of Ambrosia in 1992 with the release of Maelstrom. Can you give me a bit of history here? What led up to Maelstrom?

Andrew: I was actually in college at the time, working on a degree in Photojournalism. Maelstrom was developed over the summer, primarily because I heard someone state that good color animation wasn't possible on the technologic tour de force at the time, a Mac IIx. It served as a fun little project for me to hone my 68K asm coding skills; almost all of Maelstrom is written in assembler, with a few non-critical interface routines done in C. I wrote the game myself, but I had a number of people who assisted me, creating the graphics for the game, and all of my friends pitched in play testing it and helping me record sound effects for it. I polished it up, put it online on AOL, and was surprised at how popular it became.



Remember Maelstrom? I love the attention to detail in the graphics. The first game I remember with truly sharp 3D rendering and perfectly smooth movement.



The good guys and the bad guys.

Dave: What was the state of shareware back then? Was there any kind of movement to organize shareware? Did you get

Dave Mark is a long-time Mac developer and author and has written a number of books on Macintosh development, including *Learn C on the Macintosh*, *Learn C++ on the Macintosh*, and *The Macintosh Programming Primer* series. Be sure to check out Dave's web site at <http://www.spiderworks.com>.

a lot of payments, as compared to the number of game downloads? Where did people get most shareware? CompuServe? AOL?

Andrew: At the time, at least in my world, AOL and CompuServe were the major places to put software. The web hadn't taken off yet, and the Internet was something that only people with few social skills muttered about.

Shareware back then was fairly unorganized, in the sense that there were few products that required license codes to operate, payment was optional, and you had to send in checks or cash to pay for something. One of my favorite things was receiving money in various currencies I'd never seen before, from all over the world.

There was something distinctly cool about writing something in my dorm room in Upstate New York, and having people all over the world writing to me about it.

As for payments received vs. downloads, it's never a metric I've considered important. Many people download something, then forget about it, trash it, or whatever. I really didn't pay too much attention to the money it was making, because I was busy in college, doing things college kids do.

Dave: 1992 saw the introduction of System 7.1, as well as the Mac LC II, the Quadra 950, the 25 Mhz 68030-based PowerBook 145, the Mac IIvx and Mac Iivi, and the

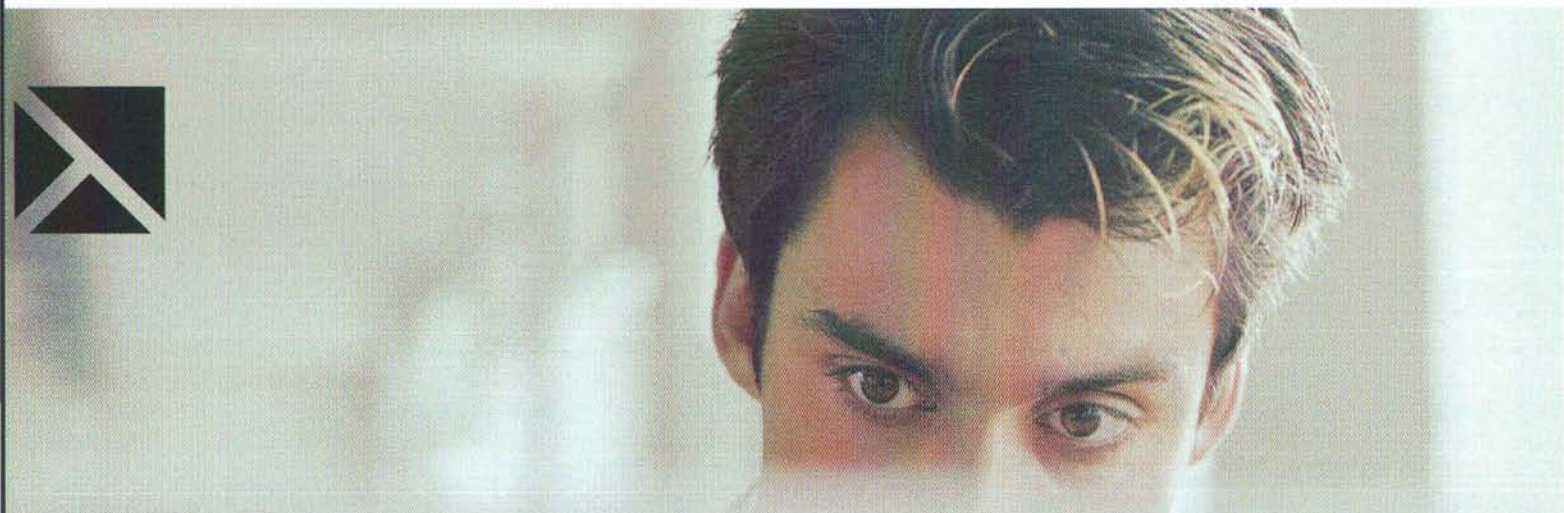
PowerBook Duo 210 and 230. Microsoft introduced Windows 3.1 and NeXT released NeXTSTEP 3.0 and NeXTSTEP 486. What was your dev environment back then?

Andrew: I actually had the original NeXT documentation, too -- I was a NeXT developer, even though I actually didn't own a machine (I couldn't afford one). On the Mac, my development environment was originally THINK Pascal, but I transitioned to using THINK C (this is before Symantec bought them out). My main development machine was a Mac Iisi at the time, with a whopping (for the time!) 17mb of RAM!

I actually started coding originally on a 512K "Fat Mac" -- yes, younguns, this was the technology of the day, and it had only 512K of memory. I used a 1200baud modem, and my life changed utterly when I was able to buy a 20mb hard drive for the machine. Prior to that, I booted from a 400K floppy drive, and did all of my development on an external 400K floppy disk.

You young whipper-snappers have no idea how good you have it! God, I feel old now...

Anyway, back to the question at hand. THINK C was a very cool development environment for the time, though it wasn't as friendly as THINK Pascal, the environment I started out programming with. I taught myself both languages, and the "bible" at the time was a phone-book style volume called "Inside Macintosh." It was mostly an



End the compromise with perfect quality and file size.

Produce highly compressed, lossless QuickTime videos for delivering screen capture training videos, animated text and graphics.

Try it FREE today at www.techsmith.com

TechSmith
ENSHARPEN[™]
Video Codec

exercise in curiosity for me; I wanted to figure out how these computers worked, and I truly enjoyed creating things.

Dave: More than ten years after Maelstrom, how has your development environment changed?

Andrew: For my personal development these days, I use Project Builder. It used to be Code Warrior, but to do some of the projects I've needed to do for Mac OS X, Project Builder was a better, if less refined, choice. But now it isn't just me; we also have an ace coder named Matt Slot working for us (we affectionately call him our "Code Whore"), as well as developers all over the world we partner with.

I don't do as much coding as I used to. I do some work on our supporting libraries, and there are some major projects that are all mine (such as Snapz Pro X), but I spent a lot of time running the company and managing projects these days. I've dabbled a bit with Cocoa, such as our free WireTap product for recording system audio, but I'm not comfortable enough with it to say I'm any good at it. I wish that I had the time to invest to dive into it deeper, but so it goes...

Other things have changed a bit. The development cycle is no longer code, run, crash, poke around in MacsBug, then reboot your computer while you slam your head into the monitor. Lather, rinse, repeat. In that sense, my sanity is significantly improved these days.

Dave: You used to do all of your development in house. How has this changed?

Andrew: We try to stay very involved in the design and development of all of the products we work on, because we collectively have a lot of experience in terms of making a polished product that people will be interested in. We are involved on many levels: helping to design the feature set/game play, offering technical assistance in the form of libraries we've written that do very useful (and hard) things, as well as debugging support, and then of course things publishers do, such as marketing, support, etc.

We do both in-house and external beta testing, phased in so that it fits with where the program is in terms of development, and we manage the beta tests. External testing is a must, because there is no other way you can get an idea how your product will perform in the wild. If I had a dollar for every time something I wrote crashed on someone else's machine, but "worked for me", I'd be rich indeed.

Dave: Clearly, games are one area where you really need to be able to move pixels quickly. Can you talk about the kinds of approaches to blitting you use in house? How about outside developers that you work with?

Andrew: We used to do all sorts of very tricky things to blit pixels to the screen quickly, such as using custom assembler for critical portions. Now it's mostly just of historical interest. Computers are fast enough now that you can safely just use the provided OS routines such as CopyBits() or even better, just use OpenGL (for both 3D and 2D).

Dave: How has your net gaming strategy evolved over the years? Is there a standard protocol for finding opponents over the net, for example? Do you roll your own net gaming libs?

Andrew: Obviously, net gaming is important, and has been for years. It's important to the point of designing your game around the physical limitations that things such as latency introduce, as well as designing your game to be a fun multiplayer game.

We looked at available solutions out there, and decided to roll our own. We use something Matt Slot wrote called Network_Tool (hey, no one said his forte was coming up with creative names), which handles TCP/IP as well as UDP networking. It performs quite well, using proprietary algorithms of concatenating and retrying transactions. This is an example of a library we develop and provide for the partners we work with to use. It takes care of the dirty, tough to write stuff, so people can get on with making their game.

Similarly, we have a library called Reggie that handles allowing people to find others on the net. It works in a very general manner, allowing for data to be stored in any format the client application wants, and it's all stored on the back end in an SQL database.

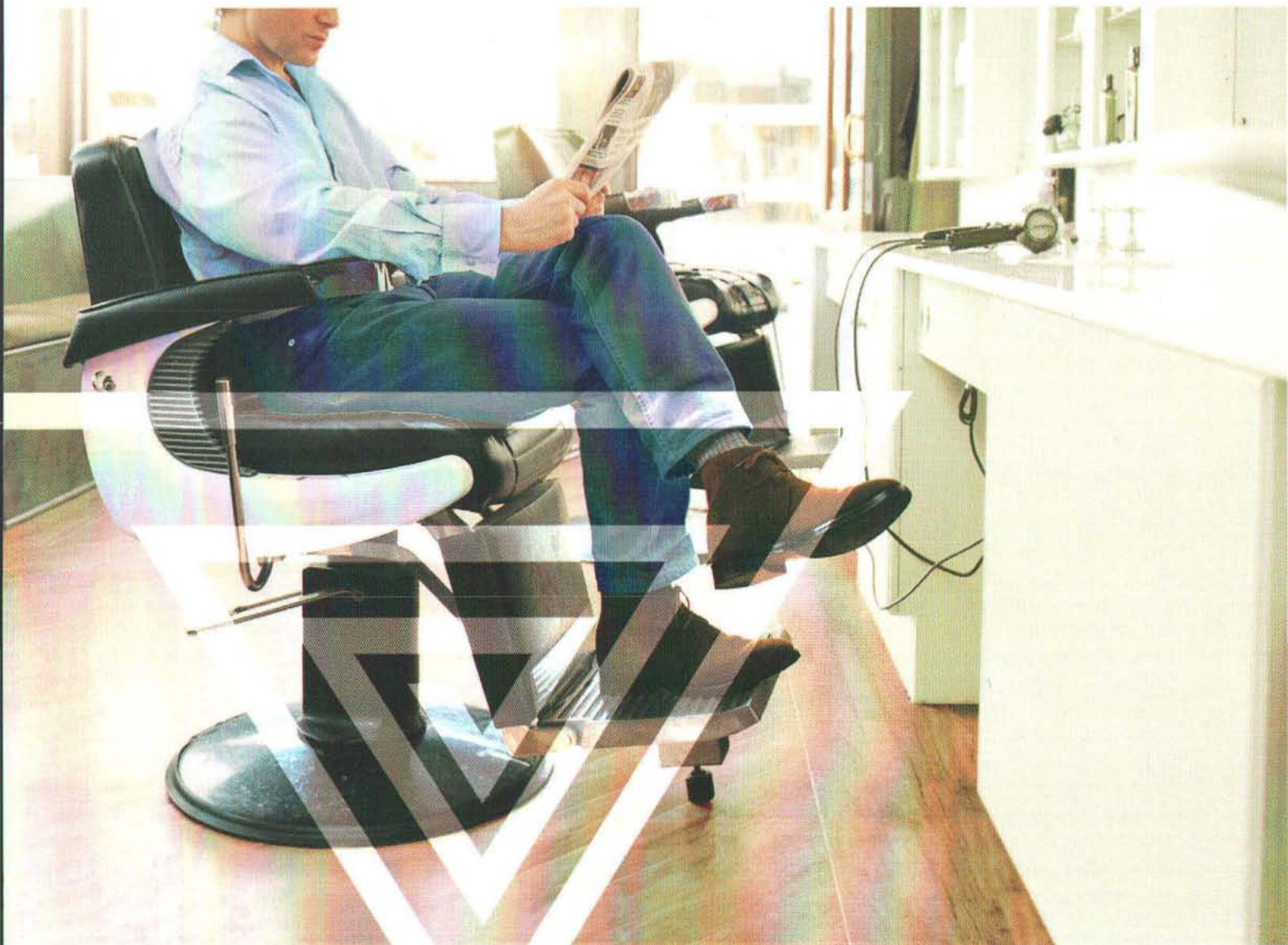
Dave: I know you recently implemented your own order processing system. How did you used to do fulfillment? How does the new system work? Is it something you wrote yourself?

Andrew: We have long had our own merchant account for credit card processing, but we decided it was long overdue that we take it to the next level and automate and enhance our order processing system. We came up with a set of design specs, and handed them off to a third party developer to implement them.

Alas, in the end, we had to do quite a bit of work ourselves to make it work the way we wanted it to. On the upside, though, our customers get their license codes instantly, they can get lost/renewed license codes sent to them any time they wish, and it all integrates seamlessly with our FileMaker Pro Server back end.

Dave: What are the technical details behind your web site? Do you use a DBMS? Is it straight HTML? Do you update it by hand? FTP? Use a management package? BBEdit?

VIRUS PROTECTION FOR MACS - EASY IN THE SOPHOS ZONE



Sophos Anti-Virus protects your organization at every level, every point of entry. Our software intelligently recognizes when files need to be virus checked, providing on-access scanning and minimizing the impact on system resources. Every license includes 24x7x365 unlimited technical support as standard and offers comprehensive protection for multiple platforms, including Mac and legacy operating systems. In the Sophos Zone, viruses don't worry Mac users at all.

GET THE FACTS:

Download a **FREE** technical brief, 'Sophos Anti-Virus for Mac OS X' from www.sophos.com/link/macbrief

www.sophos.com/link/macbrief
Tel 888-216-6703

SOPHOS
anti-virus for business

Andrew: Our web site is for the most part nothing too exciting. We have a number of custom written or modified Perl scripts that handle the drudgery of most of it. We set up systems so we generally don't have to edit HTML, just work with custom scripts we've written to do various things to it.

We update it by hand over sftp, and BBEdit is of course the tool of choice to do any kind of serious editing. It's true, it really doesn't suck.

Dave: You've got some fascinating utilities that have pushed the envelope beyond traditional application development. Snapz Pro is a perfect example. Can you take us behind the curtain on some of this technology, tell us a bit about how they work?

Andrew: Snapz Pro X is something I've been working on heavily lately. We've done a major interface reorganization, and an even more major boost in performance of the video capture engine. When 2.0 is released, we'll have made video capture as easy to do as static screen captures (something Snapz Pro X excels at already), and I'm betting that many people will opt to put a short video of their product in action up on their web site to show it off, rather than a static screen shot.

Making something like Snapz Pro X work involved very long sessions probing the various MacOS X system frameworks with nm (a command line tool) to ferret out the functionality we needed. Many times the APIs we needed didn't exist, and waiting for Apple to make them available just wasn't a viable option. Remember folks, it's software – anything is possible, it just depends on how much effort you're willing to go through to make it happen.

Dave: What advice would you give today's shareware developers?

Andrew: My main piece of advice is not to set the bar low just because they are writing shareware, or they are pricing it cheaper than commercial counterparts. People tend to judge things by why they offer them, not by how inexpensive they are. For instance, let's take games. Games are entertainment. If you go to the video store looking for a movie, you don't bypass the better movies for a budget \$2 college film. Sure, it's cheaper, but it doesn't matter, you want to be entertained.

Don't kid yourself that you can offer an inferior product just because it is cheaper, and expect it to do well.

My other main piece of advice would be to figure out what you want to do. Many people who go into think they want to do the whole shebang: development, testing, marketing, support, order processing, etc., etc. This is a quick way to lose your sanity if what you really want to do is just develop cool software. Figure out

what you really want to do, and partner with someone if it is appropriate.

Dave: One of my favorite things about being part of the Macintosh developer community is what an incredibly small world we live in. Whenever I travel I always seem to run into someone I know from this universe. Don't you have a story – something about beer?

Andrew: I recently went to Germany for Oktoberfest, which is a wild time, with some great German beers in mugs you need two hands to lift up. While sitting in an old brew house having some beers with friends, I was a little surprised to see a guy walking by with a Metrowerks t-shirt on.

So I pulled him aside, and decided to clink mugs with him, and see what he used CodeWarrior for. As it turned out, he was a Metrowerks employee... and all 20 or so people at his table were as well. So if the next version of CodeWarrior is a bit delayed, I think we may know why... the coders are all off boozing it up somewhere.

Dave: What's on the horizon for Ambrosia?

Andrew: We're always working on plenty of new projects. We just released our Internet search tool, iSeek <<http://www.AmbrosiaSW.com/utilities/iseek/>> that's going over quite well, and as I mentioned, I've had my head buried in Snapz Pro X 2.0. That's going to be a huge product when it comes out, the video capture performance is just amazing now.

Additionally, we're always working on cool new games, some of which we've announced, like our 3D racing game Redline, and others we haven't. And you can't make me talk either. :)

TILL NEXT MONTH...

I hope you enjoyed this little respite from coding. I know I did. Part of this was getting to know someone who has really been a part of the Mac gaming story. But another part of this month's column was a bit of a stroll down memory lane. I mean, c'mon, Think Pascal *before* it was bought by Symantec. I love it!!! ☺

MacTech®
M A G A Z I N E

Get MacTech delivered to your door at a price **FAR BELOW**
the newsstand price. And, it's **RISK FREE!**

Subscribe Today!

www.mactech.com

Now for Mac OS X

Ching!"
"Cha-Ching!"
"Cha-Ching!"

"Cha-Ching!"

"Cha-Ching!"



Aladdin
SECURING THE GLOBAL VILLAGE
eAladdin.com

New Privilege 6

Generate new software revenue streams — securely, effectively.

Selling, distributing and activating software electronically isn't new. But doing it securely AND all from one source certainly is news! With Privilege from Aladdin, you can have:

Via the Internet:

- A secure ESD process, transforming casual sharing into a super distribution channel for trialware
- Increased customer satisfaction and loyalty with short and simple purchasing process and resumable downloads
- Instant access to international markets with multi-language support

Via digital media and CD:

- Your choice of CD's, or pre-loaded trialware on new computers, with flexible software activation, try-before-you-buy, and try-only
- Capture new opportunities with full control over licensing terms, offers and the user's experience

Take advantage of every new revenue opportunity available today! Call us at 1-800-562-2543, or visit GoPrivilege.com to receive a FREE Privilege information pack.

PrivilegeTM
SECURE SOFTWARE eCOMMERCE

By Scott Knaster

Checking Out Kiosk Mode Features

For about 20 years now, Apple has been telling us that modes in software are bad. It's hard to remember, especially if you're a youngster, but we used to drive software around by getting into and out of modes, restricted places that provided their own definitions for the way commands worked and actions were interpreted (kind of the way a lot of venerable UNIX text editors and command line tools work in OS X today, except this was for everybody, not just us geeks). For example, when you typed a slash into a little old spreadsheet program called VisiCalc, you went into command mode, signaling that the next character was an instruction rather than something that should appear in a cell.

But Apple in its wisdom says modes are OK in certain situations. For example, modes that emulate the real world are permitted, such as picking a type of brush in a painting program. Another kind of mode is useful when you intentionally want to limit what your users can do. This rare situation becomes reality when you consider the **kiosk**, a public computer that has to fend for itself when accosted by clueless newbies, hostile geeks, and keyboard-pounding toddlers.

If you're developing an application that has to run at a kiosk, Apple helps you out by providing a set of features for **kiosk mode** in Mac OS X. In this column, we'll take a look at how to take advantage of what OS X can do for you in kiosk mode.

SIMPLE API

Apple put the cool new kiosk mode features into OS X 10.2, adding a little more in 10.3. You get to the features through two function calls: **SetSystemUIMode** and **GetSystemUIMode**. These calls let you effectively lock the user inside your application, preventing the hostile, curious, or ignorant from wreaking havoc. The **SetSystemUIMode** call is defined like this:

```
OSStatus SetSystemUIMode (SystemUIMode inMode,
                          SystemUIOptions inOptions);
```

The **inMode** parameter specifies the UI mode you want, and **inOptions** lets you choose settings for that mode. There are five

modes you can pick for your application, as shown in the following table:

System UI mode setting Details

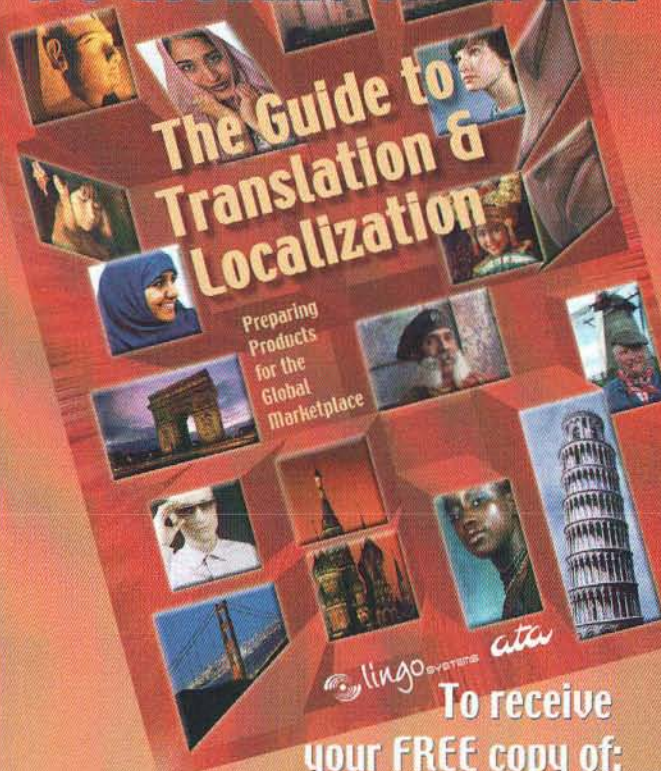
kUIModeNormal	The usual mode for all applications, and the mode you get if you don't make any calls to this API.
kUIModeContentSuppressed	This mode prevents the system from drawing anything into the content area of the screen below the menu bar unless the user performs an action that triggers an auto-show behavior. In practice, this mode simply turns dock hiding on, which makes the dock vanish until the mouse pointer moves over it, whereupon it slides onto the screen. Note: this was broken in 10.2 and is fixed in 10.3.
kUIModeContentHidden	This mode hides the dock – technically, all system UI elements other than the menu bar – and does not show it even if the user mouses into the dock region. Note: this was also broken in 10.2 and fixed in 10.3.
kUIModeAllHidden	Use this mode to hide the menu bar along with the dock, with no auto-showing feature.
kUIModeAllSuppressed	This mode was added in 10.3. Use it for hiding the dock and menu bar, but in this mode they will both auto-show if the user mouses over their content areas.

Scott Knaster has been writing about Macs for as long as there have been Macs. Scott's books *How To Write Macintosh Software* and *Macintosh Programming Secrets* were required reading for Mac programmers for more than a decade. Scott wrote developer books for General Magic and worked on Mac software for Microsoft. Scott's books have been translated into Japanese and Pascal. Scott has every issue of Mad magazine, which explains a lot.

When you call `SetSystemUIMode`, you also get to pass a set of options in addition to the mode you want to use. These options provide even more lock-in features for your kiosk and help you further refine the imprisonment of your rowdy user. Here's a list of the options you can choose:

System UI option	Details
<code>kUIOptionDisableAppleMenu</code>	Use this option to disable all items in the Apple menu, although despite the option's name, the appearance of the Apple itself isn't disabled (must be a corporate logo thing).
<code>kUIOptionDisableProcessSwitch</code>	This option turns off the user's ability to switch apps using Command-Tab and Command-Shift-Tab.
<code>kUIOptionDisableForceQuit</code>	If you select this option, the Force Quit item in the Apple menu is disabled, and Command-Option-Escape doesn't do anything. Powerful!
<code>kUIOptionDisableSessionTerminate</code>	This option helps prevent the user from shutting down or logging out. When you use it, the Restart, Shut Down, and Log Out menu items are disabled. Also, if the user presses the power button, the Restart/Sleep/Cancel/Shut Down alert won't appear.
<code>kUIOptionAutoShowMenuBar</code>	This option is only valid when used with <code>kUIModeAllHidden</code> . Pass it to make the menu bar show itself if the user rolls the mouse into its content region.
<code>kUIOptionDisableHide</code>	This option was added in 10.3. Set it to disable the Hide item in the application menu.

Classic or Cocoa Applications? We Localize Them All!



To receive
your FREE copy of:
**The Guide to Translation and
Localization - Preparing Products
for the Global Marketplace**

Call: 1-800-878-8523

Email: info@lingosys.com

Fax: 1-503-419-4873

Or visit: www.lingosys.com

**We focus on what matters most: meeting your
needs and providing excellent customer service.**

- Translation
- Desktop Publishing
- Project Management
- Localization
- Engineering
- Quality Assurance



15115 SW Sequoia Pkwy. #200 Portland, OR 97224

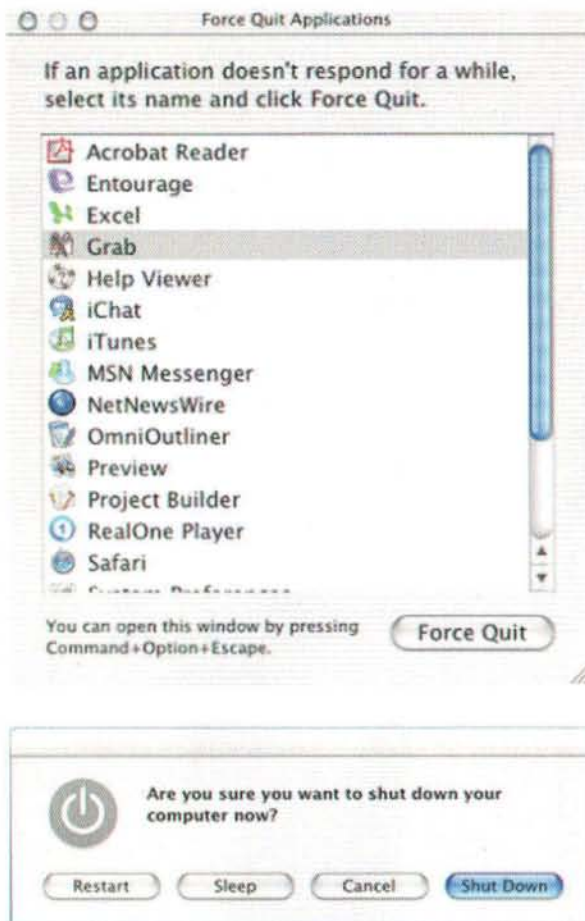


Figure 1. You can use `SetSystemUIMode` to prevent users from getting access to the Force Quit and Power windows.

One of the cautions to note in using `SetSystemUIMode` is that the mode you set is only respected when your application is frontmost. If through magical incantation or other method the user somehow manages to bring another application to the front, the UI mode for your application is switched out along with the rest of the app.

There is a corresponding call to find out the current UI mode:

```
GetSystemUIMode(SystemUIMode *outMode,
                 SystemUIOptions *outOptions);
```

As you can guess, calling `GetSystemUIMode` gets the current mode and options, in case you need to know what they are.

Depending on what your kiosk does, you might want to make the Finder go away. One good reason for this is that it prevents users from clicking on the desktop to switch out of your application. You can get rid of the Finder by sending it a `kAEQuitApplication` Apple event. Apple has an example to show you how to do that – it's at http://developer.apple.com/technotes/tn2002/downloads/tn2062_2.hqx.

If you just want your application to start up in one of the modified system UI modes and stay there, and you don't need

to set any options, you can specify your desire in the application's property list. Just add a key named `LSUIPresentationMode` with type number and a value of 0, 1, 2, 3, or 4. Use 0 for normal mode, 1 for content suppressed, 2 for content hidden, 3 for all hidden, and 4 for all suppressed.

To help you view the power of the kiosk mode APIs in all their majesty, Apple provides a cool sample app, called `UsingSystemUIMode`, that demonstrates the various available features and tweaks. See **Figure 2** for a shot of the app's screen.



Figure 2. Apple's sample application for testing kiosk mode APIs.

To get the sample app and to learn more about implementing kiosk mode in your application, see Technical Note TN2062, Guide to Creating Kiosks on Mac OS X, available at <http://developer.apple.com/technotes/tn2002/tn2062.html>.

NO CAN DO

Although the kiosk mode API is a good start toward locking up your kiosk machine, there are some things you'll want to do that you can't do yet. And just when you thought everything was perfect. Here's more information about some of those limitations:

You can't disable the eject key. If your kiosk has to run with a CD or DVD inserted, you can design the kiosk so that it's physically impossible to eject the disk.

To disable various wacky scenarios, such as taking control of the machine by booting off an external disk or booting into single-user mode by holding down Command-S, you can enable the Mac's Open Firmware password. When the Open Firmware password is enabled, the Mac will only boot from the startup disk you select in System Preferences.

You can't prevent the user from messing with the brightness keys and darkening the screen, nor can you intercept the volume keys.

ONE MORE THING

Needless to say, locking your users in and preventing them from having basic features is not a standard design principle. In a kiosk, many of these features are necessary. In a typical app, they're practically criminal. Please use them with care, or at least a sense of humor.

**"Without a doubt, the Premiere Resource Editor
for the Mac OS ... A wealth of time-saving tools."**

— MacUser Magazine Eddy Awards

"A distinct improvement over Apple's ResEdit."

— MacTech Magazine

"Every Mac OS developer should own a copy of Resorcerer."

— Leonard Rosenthol, Aladdin Systems

**"Without Resorcerer, our localization efforts would look like a
Tower of Babel. Don't do product without it!"**

— Greg Galanos, CEO and President, Metrowerks

"Resorcerer's data template system is amazing."

— Bill Goodman, author of *Smaller Installer* and *Compact Pro*

"Resorcerer Rocks! Buy it, you will NOT regret it."

— Joe Zobkiw, author of *A Fragment of Your Imagination*

"Resorcerer will pay for itself many times over in saved time and effort."

— MacUser review

"The template that disassembles 'PICT's is awesome!"

— Bill Steinberg, author of *Pyro!* and *PBTools*

"Resorcerer proved indispensable in its own creation!"

— Doug McKenna, author of *Resorcerer*



Resorcerer® 2

Version 2.0

The Resource Editor for the Mac™ OS Wizard

ORDERING INFO

Requires System 7.0 or greater,
1.5MB RAM, CD-ROM

Standard price: \$256 (decimal)

Website price: \$128 - \$256

(Educational, quantity, or
other discounts available)

Includes: Electronic documentation
60-day Money-Back Guarantee
Domestic standard shipping

Payment: Check, PO's, or Visa/MC
Taxes: Colorado customers only

Extras (call, fax, or email us):
COD, FedEx, UPS Blue/Red,
International Shipping

MATHEMAESTHETICS, INC.
PO Box 298
Boulder, CO 80306-0298 USA
Phone: (303) 440-0707
Fax: (303) 440-0504
resorcerer@mathemaesthetics.com

New
in
2.0:

- Very fast, HFS browser for viewing file tree of all volumes
- Extensibility for new Resorcerer Apprentices (CFM plug-ins)
- New AppleScript Dictionary ('aete') Apprentice Editor
- MacOS 8 Appearance Manager-savvy Control Editor
- PowerPlant text traits and menu command support
- Complete AIFF sound file disassembly template
- Big-, little-, and even mixed-endian template parsing
- Auto-backup during file saves; folder attribute editing
- Ships with PowerPC native, fat, and 68K versions

- Fully supported; it's easier, faster, and more productive than ResEdit
- Safer memory-based, not disk-file-based, design and operation
- All file information and common commands in one easy-to-use window
- Compares resource files, and even **edits your data forks** as well
- Visible, accumulating, editable scrap
- Searches and opens/marks/selects resources by text content
- Makes global resource ID or type changes easily and safely
- Builds resource files from simple Rez-like scripts
- Most editors DeRez directly to the clipboard
- All graphic editors support screen-copying or partial screen-copying
- Hot-linking Value Converter for editing 32 bits in a dozen formats
- Its own 32-bit List Mgr can open and edit very large data structures
- Templates can pre- and post-process any arbitrary data structure
- Includes nearly 200 templates for common system resources
- TMPLs for Installer, MacApp, QT, Balloons, AppleEvent, GX, etc.
- Full integrated support for editing color dialogs and menus
- Try out balloons, 'ictb's, lists and popups, even create C source code
- Integrated single-window Hex/Code Editor, with patching, searching
- Editors for cursors, versions, pictures, bundles, and lots more
- Relied on by thousands of Macintosh developers around the world

To order by credit card, or to get the latest news, bug fixes, updates, and apprentices, visit our website...

www.mathemaesthetics.com

Andrew Troelsen, Minneapolis, Minnesota

Compiling, disassembling and re-assembling .NET binaries

Exploring .NET development on Mac OS X

DIVING DEEPER INTO THE .NET ASSEMBLY FORMAT

The previous issue got you up and running with the SSCLI, and introduced you to the basics of the C# compiler (csc) and CLI execution launcher (clix) utilities. In this installment, you will not only come to learn some additional options of csc, but also examine two key development tools (ildasm and ilasm) which any .NET enthusiast must be aware of. Along the way, you will gain a much deeper understanding how a .NET assembly is composed under the hood.

A REVIEW OF ASSEMBLY BASICS

During the course of these first few installments, you have come to learn that the .NET platform supports a high degree of binary reuse, which is made possible using a unit of deployment termed an *assembly*. Recall that .NET assemblies (which may take an *.exe, *.dll or *.netmodule file extension) are composed of three key elements: CIL code, type metadata and the assembly manifest:

- **Common Intermediate Language (CIL):** The true language of the .NET platform. All .NET-aware compilers transform their respective tokens into the syntax of CIL. As noted earlier, CIL code is conceptually similar to Java bytecode in that CIL is platform agnostic. Unlike Java bytecode however, CIL is compiled (not interpreted) on demand by the .NET runtime.
- **Type Metadata:** Used to fully describe each and every aspect of a .NET type (class, interface, structure, enum or delegate) referenced by, or defined within, a given assembly. As you get to know the .NET platform, you will

quickly discover that type metadata is the glue for every major technology.

- **Manifest:** Metadata that describes the assembly itself (version, required external assemblies, copyright information and so forth). As described at a later time, the assembly manifest is a key aspect of the .NET versioning policy.

The good news is that you will never have to author a lick of CIL code, type metadata or manifest information by hand, as it is the job of a .NET compiler to translate your source code into the correct binary format. Given this, you will never need to describe your types manually (as is the case in CORBA IDL) or build C-based wrapper classes to communicate with an underlying protocol. While the .NET platform supports multiple languages and corresponding compilers, few would argue that C# is the language of choice used to build .NET assemblies.

In the last issue, you created a .NET code library (MyLib.dll) and a corresponding .NET client executable (MyClient.exe). While you are free to reuse that source code over the course of this issue, **Listings 1** and **2** offer a few simple types to manipulate over the pages that follow.

Listing 1: simpleCodeLibrary.cs

```
//A simple code library containing one class type.
using System;

namespace ShamerLib
{
    public class Shamer
    {
        public static void ShameChild(string kidName,
            int intensity)
        {
            for(int i = 0; i < intensity; i++)
                Console.WriteLine("Be quiet {0}!!", kidName);
        }
    }
}
```

Andrew Troelsen is a seasoned .NET developer who has authored numerous books on the topic, including the award winning *C# and the .NET Platform*. He is employed as a full-time .NET trainer and consultant for Intertech Training (www.intertechtraining.com), and is a well-known Timberwolves, Wild and Vikings rube (not necessarily in that order). You can contact Andrew at atroelsen@mac.com.



Bob and Sue use a MAC for business,
but their accountant uses a PC.
Do you know what kind of
accounting software they need?

Hmmm. That could be a problem, but not with MYOB.

Despite their different operating systems, Bob and Sue and their very clever accountant, Nick The Numbers Man, are able to manage finances, create reports, and produce documents as easy as can be.

MAC or Windows—it makes no difference, they don't even have to be in the same place at the same time.

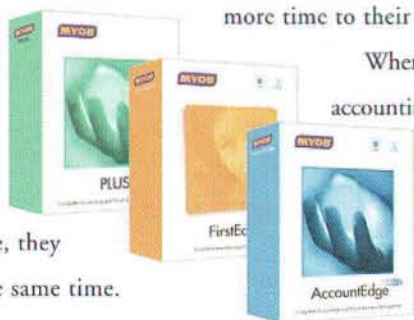
They can do it all by e-mail so quickly that they're able to devote more time to their favorite customers and clients.

Wherever you are, whatever the platform, MYOB is the accounting software that's a smart choice for everyone. It turns everyday accounting and financial management into something that's almost fun.

**MYOB,
THE SMALL BUSINESS ANSWER.**

©2003 MYOB US, Inc.

(800)322-MYOB www.myob.com/us



Listing 2: simpleClientApplication.cs

```
// Our client application makes use
// of Shamer.dll (seen in Listing 1).
using System;
using ShamerLib;

namespace ShamerClient
{
    public class TheApp
    {
        public static void Main(string[] args)
        {
            string kidName;
            int frustrationLevel;

            // All .NET arrays derive from System.Array.
            // This class type has a member named Length.
            if (args.Length != 0)
            {
                kidName = args[0];
                frustrationLevel = int.Parse(args[1]);
            }
            else // No command line args, prompt user.
            {
                Console.WriteLine("Please enter child's name: ");
                kidName = Console.ReadLine();
                Console.WriteLine("Please enter annoyance level: ");
                frustrationLevel = int.Parse(Console.ReadLine());
            }

            // Pass user input to Shamer type.
            Shamer.ShameChild(kidName, frustrationLevel);
            Console.WriteLine("Thanks for playing...");
        }
    }
}
```

The only real point of interest in **Listing 2** is the fact that we have provided the ability to process command line arguments via the incoming array of strings. If the caller does supply such arguments, we will use them to assign our local variables (kidName and frustrationLevel). On the other hand, if the user does not supply command line arguments, s/he will be prompted at the Terminal. In either case, once the local variables have been assigned, they are passed into the static ShamerLib.Shamer.ShameChild() method located in the shamer.dll assembly.

BUILD DETAILS OF THE C# COMPILER

Now that we have some C# source code at our disposal, let's examine select details of the SSCLI C# compiler, csc. The C# compiler (csc) is a command line tool which supports numerous arguments, the full set of which can be viewed by typing the command shown in **Listing 3** from an SSCLI-enabled Terminal (see the previous issue for details regarding sourcing the SSCLI environment variables):

Listing 3: Viewing each option of csc.

```
csc -help
```

Now, assume you wish to compile simpleCodeLibrary.cs into a single file assembly named shamer.dll. To do so, you need to be aware of the following core commands of csc.exe:

- **/target:** Allows you to specify the format of the output file (code library, Terminal application and so on). May be shortened to /t.

- **/out:** Allows you to specify the name of the output file. If omitted, the name of the output file is based on the name of the class which contains the Main() method, or in the case of a class library, the name of the initial input file.
- **/reference:** Allows you to specify any external assemblies to reference during the current compilation (may be shortened to /r).

At it's core, csc processes C# source code files (which by convention take a *.cs file extension) to produce a binary file termed an assembly. Under the SSCLI, csc provides three valid flags that control the target output (**Table 1**).

Option of /target flag	Short Form	Meaning in Life
/target:exe	/t:exe	Generates an executable Terminal assembly with the *.exe file extension.
/target:dll	/t:dll	Generates a binary code library that may not be directly executed by the .NET runtime. Rather, *.dll binaries are loaded by an executing *.exe.
/target:module	/t:module	This option allows you to build a 'multi-file' assembly. In a nutshell, multi-file assemblies are a collection of code library files that are intended to be versioned as a single logical unit. This particular aspect of .NET will be examined at a later time.

Table 1. Variations of the /target flag

In addition to the /t:exe, /t:dll and /t:module files, the csc compiler of the SSCLI also supplies the /target:winexe flag, which effectively does nothing under the Macintosh operation system (under GUI-aware implementations of the C# compiler, /t:winexe is used to build a GUI-based Windows Forms application).

Of course you will also need to specify the set of input files to be processed. These can be listed individually as discrete arguments, or via the wildcard syntax (e.g., csc *.cs) to instruct csc.exe to compile all C# files in the current directory. However, to build shamer.dll, the following command set will do nicely (**Listing 4**):

Listing 4: Compiling the Shamer.dll code library.

```
csc /t:library /out:shamer.dll simpleCodeLibrary.cs
```

To compile simpleClientApplication.cs into a single file executable assembly that makes use of the types contained within shamer.dll, enter the command shown in **Listing 5**

(be sure that shamer.dll is in the same location of the input *.cs file):

Listing 5. Compiling the client.exe executable.

```
csc /t:exe /out:client.exe /r:shamer.dll
simpleClientApplication.cs
```

At this point you are able to run the resulting client.exe application (either with or without command line arguments) using the clx utility (Listing 6).

Listing 6. Executing your .NET example application.

```
clx client.exe Sally 9
```

Here we are passing two arguments to the client.exe application. Recall that we have programmed Main() to parse these values out of the incoming array of strings, and therefore the phrase "Be quite Sally!!!" will print out nine times. If you run client.exe without specifying any command line arguments, you will be prompted at the Terminal for further input. Do note that the version of Main() is not smart enough to receive these arguments out of order (just to keep focused on the tasks at hand). Given this, if you specify the child's name after the intensity level, you will receive a runtime exception.

Working with C# Response Files

When you work with the C# command line compiler, you are likely to issue the same options during each compilation cycle. For example, assume you have a directory which contains ten *.cs files, of which you wish to compile three of them into a single file .NET class library named MyLib.dll. Furthermore, assume you need to reference various external assemblies for the build (Listing 7):

Listing 7. A non-trivial csc command set.

```
csc /out:MyLib.dll /t:library /r:System.Xml.dll
/r:MyCoolLib.dll /r:MyOtherLib.dll firstClass.cs
otherClass.cs thirdClass.cs
```

Clearly, this can become a pain. To help make your Terminal compilations more enjoyable, the C# compiler supports the use of 'response files'. These text files (which by convention take an *.rsp file extension) contain all of the options you wish to pass into csc. Assume you have created a response file (Listing 8) named compilerArgs.rsp (note that comments are denoted with the '#' symbol):

Listing 8. CompilerArgs.rsp.

```
# compilerArgs.rsp.
# Name of output file.
/out:MyLib.dll

# Type of output file.
/t:library

# External assemblies to reference.
/r:System.Xml.dll
/r:MyCoolLib.dll
/r:MyOtherLib.dll
```

```
# Input files.
firstClass.cs otherClass.cs thirdClass.cs
```

Given this, you can now simply pass in the compilerArgs.rsp file as the sole argument to csc.exe via the '@' flag (Listing 9):

Listing 9. Specifying a response file at the command line.

```
csc @compilerArgs.rsp
```

If you wish, you may feed in multiple response files (Listing 10):

Listing 10. csc may be fed in multiple response files.

```
csc @compilerArgs.rsp @moreArgs.rsp @evenMoreArgs.rsp
```

Keep in mind however that response files are processed in the order they are encountered; therefore settings in a previous file can be overridden by settings in a later file. Finally, be aware that there is a 'default' response file, csc.rsp, which is automatically processed by csc.exe during each compilation. If you examine the contents of this file (which is located by default under /sscli/build/v1.ppcfsthk.rotor) you will find little more than a set of common assembly references (such as System.dll, System.Xml.dll and so on). In the rare occasion that you wish to disable the inclusion of csc.rsp you may specify the /noconfig flag.

high quality - competitive rates - 16 years experience - award winning

reliable - high quality - competitive rates - efficient - award winning - qa services - reputable

Full Spectrum Software

Development & Testing

Device Drivers

CarbonCocoa

Real Basic

Rescue Missions

Cross Platform Development

1661 Worcester Road
Framingham, MA 01701
508-620-6400
www.FullSpectrumSoftware.com

competitive rates - 16 years experience - award winning - high quality

Of course, the C# compiler defines many other options beyond the set we have just examined. Over the life of the series, you'll see additional options at work, however check out </sscli/docs/compilers/csharp.html> for all the gory details of the csc utility which ships with the SSCLI.

THE ROLE OF THE CIL DISSASSEMBLER (ILDASM)

At this point you should be confident in your ability to build and debug .NET assemblies using csc (of course, C# itself may still be a bit of a mystery, but we'll deal with that soon enough). The next topic to address is the process of getting under the hood of a binary .NET blob and checking out the key ingredients (CIL, type metadata and manifest information). The SSCLI ships with numerous programmer tools to do this very thing, the first of which is the Intermediate Language Disassembler (ildasm). **Table 2** lists some of the key options of this (very insightful) utility (see </sscli/docs/tools/ildasm.html> for full details).

Option of ildasm Meaning in Life

/classlist	Displays a list of all types within a given .NET assembly.
/item	This option allows you to view the disassembly of a specific type or type member, rather than the entire set of type members.
/metadata	Use this flag to view the type metadata within the .NET assembly.
/output:<filename>	Instructs ildasm to dump the contents to a specified text file, rather than to the Terminal.
/visibility	This flag instructs ildasm to display types of a certain 'visibility' level. In a nutshell, C# supports the creation of public types (which may be used by other assemblies) and internal types (which may only be used by the defining assembly). The CIL programming language defines a number of additional visibility modifiers.

Table 2. Some (but not all) options of ildasm.

To take this tool out for a test drive, let's view the internal structure of the entire client.exe assembly (**Listing 11**).

ildasm client.exe

Listing 11. Disassembling client.exe via ildasm

Viewing the Assembly Manifest

The first point of interest is the manifest data, which as you recall is metadata that defines the assembly itself (**Listing 12**).

```
.assembly extern mscorlib
{
    .publickeytoken = (B7 7A 5C 56 19 34 E0 89 )
    .ver 1:0:3300:0
}
[assembly extern shamer
{
```

Listing 12. The manifest data of client.exe

```
.ver 0:0:0:0
}
[assembly client
{
    .hash algorithm 0x00008004
    .ver 0:0:0:0
}
.module client.exe
.imagebase 0x00400000
.subsystem 0x00000003
.file alignment 512
.corflags 0x00000001
```

The critical point here is the use of the .assembly extern directives. These CIL token are used to document the set of external assemblies the current assembly must have to function correctly. Notice that client.exe makes use of the standard mscorlib.dll assembly as well as (surprise, surprise) shamer.dll. Next, notice that the .assembly directive (without the extern attribute) is used to describe basic characteristics of the current assembly (client.exe in this case) such as it's version (.ver) and module name (.module). As mentioned, the .NET runtime reads this information during the process of locating and loading external assemblies for use (exactly *how* is the topic of a later article).

Viewing the CIL behind Main()

Next, let's examine the CIL code produced by csc for the client's static Main() method (**Listing 13**). Be aware that each of the IL_XXXX: tokens are line-labels inserted by ildasm.

```
.method public hidebysig static void
    Main(string[] args) cil managed
{
    .entrypoint
    .maxstack 2
    .locals init (string V_0, int32 V_1)
    IL_0000: ldarg.0
    IL_0001: ldlen
    IL_0002: conv.i4
    IL_0003: brfalse.s IL_0014
    IL_0005: ldarg.0
    IL_0006: ldc.i4.0
    IL_0007: ldelem.ref
    IL_0008: stloc.0
    IL_0009: ldarg.0
    IL_000a: ldc.i4.1
    IL_000b: ldelem.ref
    IL_000c: call int32 [mscorlib]
        System.Int32::Parse(string)
    IL_0011: stloc.1
    IL_0012: br.s IL_0039
    IL_0014: ldstr "Please enter child's name: "
    IL_0019: call void [mscorlib]
        System.Console::Write(string)
    IL_001e: call string [mscorlib]
        System.Console::ReadLine()
    IL_0023: stloc.0
    IL_0024: ldstr "Please enter annoyance level: "
    IL_0029: call void [mscorlib]
        System.Console::Write(string)
    IL_002e: call string [mscorlib]
        System.Console::ReadLine()
    IL_0033: call int32 [mscorlib]
        System.Int32::Parse(string)
    IL_0038: stloc.1
    IL_0039: ldloc.0
    IL_003a: ldloc.1
    IL_003b: call void [shamer]
```

Listing 13. The CIL of Main().

Multiple formats. Multiple platforms. Complex installers.

Aladdin solves the compression and installation puzzle.

**Trying to figure out how to handle multiple
compression formats and platforms?**

The StuffIt Engine solves the compression puzzle.



Aladdin's StuffIt Engine SDK:

- Adds value to your application by integrating powerful compression and encryption.
- Is the only tool that supports the Stuffit file format.
- Provides a single API that supports over 20 compression and encoding formats common on Macintosh, Windows, and Unix.
- Makes self-extracting archives for either Macintosh or Windows.
- Available for Macintosh, Windows, Linux, or Solaris.

Licenses start as low
as \$99/year

To learn more, visit:
www.stuffit.com/sdk/

Stuffit Engine SDK™ The power of Stuffit in your software.



**Looking for the easiest and fastest
way to build an installer?**

Stuffit InstallerMaker completes your puzzle.

It's not enough just to write solid code anymore. You still have to write an installer for your users. Stuffit InstallerMaker makes it simple and effective.

- Stuffit InstallerMaker gives you all the tools you need to install, uninstall, resource-compress or update your software in one complete, easy-to-use package.
- Add marketing muscle to your installers by customizing your electronic registration form to include surveys and special offers.
- Make demoware in minutes. Create Macintosh OS X and Macintosh Classic compatible installers with Stuffit InstallerMaker.

Prices start at \$250

To learn more, visit:
[www.stuffit.com/
installermaker/](http://www.stuffit.com/installermaker/)

Stuffit InstallerMaker™ The complete installation solution.™



www.stuffit.com
(831) 761-6200

© 2002 Aladdin Systems, Inc. Stuffit, Stuffit InstallerMaker, and Stuffit Engine SDK are trademarks of Aladdin Systems, Inc. The Aladdin logo is a registered trademark. All other products are trademarks or registered trademarks of their respective holders. All Rights Reserved.


```

    ShamerLib.Shamer::ShameChild(string, int32)
IL_0040: ldstr "Thanks for playing..."
IL_0045: call void [mscorlib]
    System.Console::WriteLine(string)
IL_004a: ret
// end of method TheApp::Main

```

Now, before your eyes pop out of your head, let me reiterate that this is not the place to dive into all the details of the syntax of CIL. However, here is a brief explanation of the highlights.

First of all, CIL is an entirely stack based language: values are pushed onto the stack using various CIL operational codes (opcodes) and popped over the stack using others. Notice that the `Main()` method is adorned with the `.entrypoint` directive, which as you may guess is how the .NET runtime is able to identify entry point to the executable.

Next, note that the `.maxstack` directive is used to mark the upper limit of values which may be pushed onto the stack during the duration of this method. Given that our `Main()` method defines only two local variables and never calls methods with more than two arguments, it should be no surprise the value assigned to `.maxstack` is 2.

Now, notice the syntax used to invoke a type method (**Listing 14**).

Listing 14. Invoking `Console.WriteLine()` and `Shamer.ShameChild()`.

```

IL_0019: call void [mscorlib]
    System.Console::Write(string)
...
IL_003b: call void [shamer]
    ShamerLib.Shamer::ShameChild(string, int32)

```

As you can see, the call opcode marks the act of invoking a method. However, notice that the friendly name of the assembly is infixed between the return value and method name. Given this, we can boil down a CIL method invocation to the following simple template shown in **Listing 15**:

Listing 15. The CIL of method invocations.

```

ReturnType[NameOfAssembly]
Namespace.Type::MethodName(anyArguments)

```

Again, don't sweat the details of CIL at this point in the game (in fact, you can live the life of a happy and healthy C# developer without thinking about CIL whatsoever). Simply understand that the `ildasm` tool allows you the raw CIL syntax emitted by a given .NET compiler.

Viewing the Type Metadata

Recall that an assembly is composed of three key elements (CIL, type metadata and manifest information). If you wish to view the type metadata within a given assembly, you must specify the `/metadata` option (**Listing 16**).

Listing 16. Listing type metadata with `ildasm`

```
ildasm /metadata client.exe
```

Once you execute this command, you will find that the initial output of `ildasm` is indeed a listing of type metadata. Simply put, .NET metadata can be lumped into two categories: `TypeDefs` (types you defined yourself) and `TypeRefs` (types within an external assembly you referenced). Both of these categories will list in vivid detail the composition of each item (e.g. base class, number of methods, method parameters, etc). For example, **Listing 17** shows the metadata for the `ShamerClient.TheApp` class type defined within `client.exe` (again, don't sweat the details).

Listing 17. Type metadata for `TheApp`

```

// TypeDef #1
// -----
// TypDefName: ShamerClient.TheApp (02000002)
// Flags : [Public] [AutoLayout]
// [Class] [AnsiClass] (00100001)
// Extends : 01000001 [TypeRef] System.Object
// Method #1 [ENTRYPOINT]
// -----
// MethodName: Main (06000001)
// Flags : [Public] [Static]
// [HideBySig] [ReuseSlot] (00000096)
// RVA : 0x00002050
// ImplFlags : [IL] [Managed] (00000000)
// CallConvntn: [DEFAULT]
// ReturnType: Void
// 1 Arguments
// Argument #1: SZArray String
// 1 Parameters
// (1) ParamToken : (08000001) Name :
// args flags: [none] (00000000)
//
// Method #2
// -----
// MethodName: .ctor (06000002)
// Flags : [Public] [HideBySig] [ReuseSlot]
// [SpecialName] [RTSpecialName] [.ctor] (00001886)
// RVA : 0x000020a8
// ImplFlags : [IL] [Managed] (00000000)
// CallConvntn: [DEFAULT]
// hasThis
// ReturnType: Void
// No arguments.

```

Now, you may be wondering what use type metadata serves in the .NET platform. To be honest, just about *everything* in .NET revolves around metadata in one form or another. Object serialization, XML Web services, .NET remoting, late binding, dynamic type creation and heap allocations all demand full type descriptions. Later in this series, you will learn how to leverage this information programmatically using the friendly object model provided by the `System.Reflection` namespace (can anyone say *custom object browser*?)

THE METAINFO UTILITY

On a metadata-related note, the SSCLI supplies an additional tool (metainfo) which is used exclusively to view type metadata (see `/sscli/docs/tools/metainfo.html` for full details). For example, if you wish to see the `TypeDefs` and `TypeRefs` within `shamer.dll` (but don't care to see the CIL code or manifest data), you could enter the following command (**Listing 18**):

Listing 18. Working with the metainfo utility.

```
metainfo shamer.dll
```


THE ROLE OF THE CIL ASSEMBLER (ILASM)

Now that you can disassemble a .NET assembly using ildasm and metainfo, it is worth pointing out that the SSCLI ships with a CIL assembler utility named (not surprisingly) ilasm. Although it is not terribly likely, it is entirely possible to build an complete .NET application using raw CIL code and bypass higher level languages such as C#, VB.NET and so forth (remember, as far as the .NET runtime is concerned, it's *all* CIL). As suggested by **Table 3**, working with ilasm is quite straightforward.

Options of ilasm Meaning in Life

/clock	Tells ilasm to display compilation diagnostics for the current build.
/dll or /exe	Builds a code library or executable assembly (respectively).
/output	Specifies the name of the output file.

Table 3. Some (but not all) options of ilasm.

As a fellow programmer, I'm sure you'd love to build a .NET assembly using CIL and the CIL assembler at least once (just to say you did it). Again, building anything but a trivial CIL source code file would require a solid understanding of the syntax and semantics of the Common Intermediate Language, however if you are up for the task, create a brand new source code file named simpleCILCode.il (by convention, CIL code files take an *.il extension). Within your new file, define the following .NET type (**Listing 19**):

Listing 19. An example using raw CIL

```
// mscorlib.dll is automatically
// listed in the manifest by ilasm,
// so we don't need to bother specifying
// this external assembly.

// Now defined our assembly.
// If unspecified, the version
// of an assembly is automatically
// 0.0.0.0.
.assembly SimpleCILCode[]
.module SimpleCILCode.dll

// Our only class type: MyCILExample.MyCILApp
.namespace MyCILExample
{
    .class public auto ansi beforefieldinit MyCILApp
    extends [mscorlib]System.Object
    {
        //The single method, Speak().
        .method public hidebysig static void
        Speak() cil managed
        {
            .maxstack 1
            ldstr "Yo!!"
            call void [mscorlib]
            System.Console::WriteLine(string)
            ret
        }
    }
}
```

Basically, this CIL code file defines a single namespace (MyCILExample) which contains a single class (MyCILApp) which supports a single method named Speak(). The implementation of Speak() loads a string literal onto the stack (via the ldstr opcode) which is used for the invocation of System.Console.WriteLine(). The ret opcode, obviously, returns from the method. Now, to compile this CIL source code file into

REALbasic

AppleScript

Director

Java

SQL

C++

Revolution

XCMD

MetaCard

SuperCard

WebSiphon

futureBASIC

VisualBasic

Valentina

Object-Relational SQL Database

The fastest database engine
for MacOS/Windows

It operates 100's and sometimes
a 1000 times faster than other systems

www.paradigmasoft.com

Hosted by macserve.net

Download full featured evaluation version

a binary *.dll, supply the following command set to ilasm (**Listing 20**):

Listing 20. Compiling *.il files using ilasm

```
ilasm /output: Simple.dll /dll SimpleCILCode.il
```

At this point, you can make use of Simple.dll just as you would any other .NET code library. To prove the point, let's update our existing simpleClientApplication.cs file to invoke the Speak() method (**Listing 21**).

Listing 21. Our updated client application

```
using System;
using ShamerLib;

// Need this!
using MyCILExample;

namespace ShamerClient
{
    public class TheApp
    {
        public static void Main(string[] args)
        {
            ...
            MyCILApp.Speak();
        }
    }
}
```

Now recompile client.exe while referencing simple.dll (**Listing 22**).

Listing 22. Recompiling client.exe

```
csc /r:simple.dll /r:shamer.dll /t:exe /out:client.exe
simpleClientApplication.cs
```

Sure enough, if you run simpleClientApplication.exe through ildasm, you will find Simple.dll listed in the assembly manifest. Likewise, if you run the updated application, you will find that the message ("Yo!!") is emitted to the Terminal.

ROUND TRIPPIN' (ASSEMBLY TO CIL, CIL TO ASSEMBLY)

Given the functionality of ildasm and ilasm, the SSCLI (as well as other .NET platform distributions) intrinsically supports the notion of a 'round trip'. Simply put, this software idiom is used to describe the process of compiling -> decompiling -> editing -> recompiling a software blob into a new modified unit. As you would guess, this can prove extremely helpful when you need to modify the contents of a .NET assembly to which you do not have the original source code files. To solidify the information presented in this issue, try the following round-trip exercise:

- Disassemble your client application using ildasm, outputting the CIL to a new file named simpleClientApplication.il (don't forget the /output flag of ildasm).

- Using your text editor of choice, modify each string literal (e.g. "Please enter child's name") to a new string ("Yo dude! Enter the name of the kid!").
- Save your changes and recompile the *.il file into a new executable assembly named ModifiedApp.exe using ilasm.
- Run your ModifiedApp.exe assembly using clix.

A Brief Note Regarding Obfuscation

Now obviously, if a .NET assembly can be so easily disassembled, modified and recompiled, you are no likely already imagining numerous doomsday scenarios (your proprietary 'bubble sort' algorithm has been modified to wipe a user's hard-drive of all data) and copyright infringements ("But that is *our* CIL code, you can't change it!") Yes it is true, given that a .NET binary can always be viewed in terms of its CIL code using tools such as ildasm, it is possible that prying eyes could take your intellectual property as a basis for their own and build a software monster. This problem is not unique to the .NET platform however, as numerous Java, C(++) and BASIC decompilers have existed for years.

However, if you wish to lessen the chances of bad people using your resulting CIL code for evil purposes, rest assured that numerous .NET obfuscators exist. As you may know, the basic role of an obfuscator is to make use of a set of algorithms translate valid syntax (in this case CIL code) into and out-of total nonsense. As well, the .NET platform supports the notion of a 'strong name' which can be used to digitally sign (in effect) your assemblies using public / private key cryptography. By doing so, it is next to impossible for an evil individual to modify a binary assembly and pretend to 'be you'. More on strong names at a later time.

WRAP UP

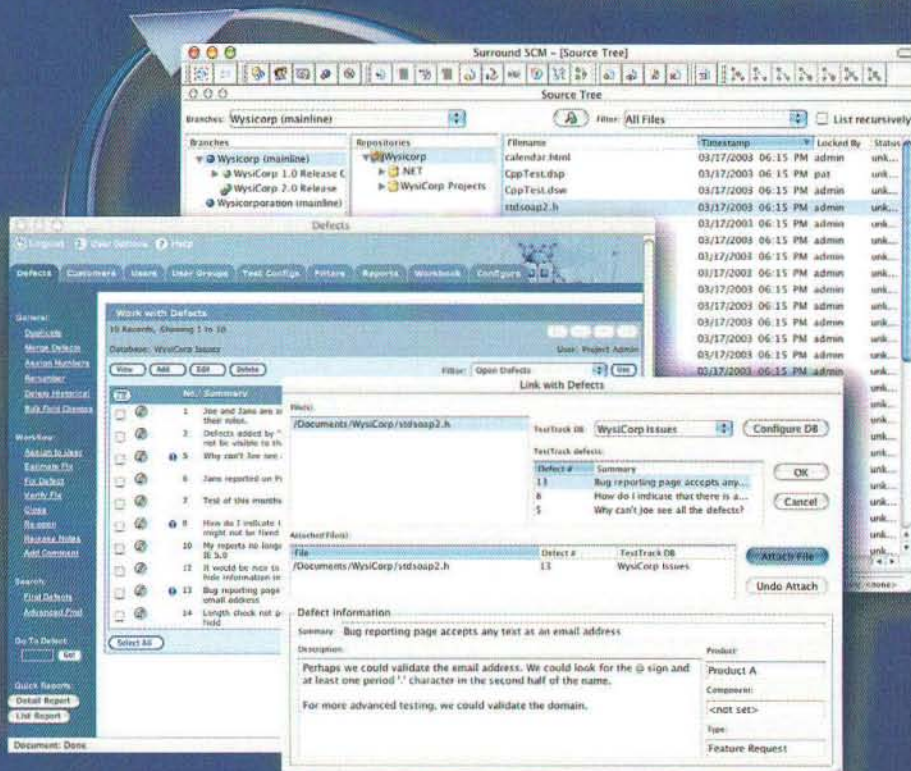
Sweet! So here ends another installment of Casting your .NET where you spent your time digging deeper into the format of a .NET assembly. We began by revisiting the key elements of a .NET binary (CIL, metadata and manifest information) and learned about some additional functionality provided by csc. Next up, you came to understand the role of the ildasm utility and learned how this tool allows .NET developers to peek inside the assembly itself to view the underlying goo. Finally, you took ilasm out for a test drive and preformed a simple 'round-trip'.

In the next issue, you will finalize your initial look at the SSCLI and examine a number of interesting sample applications, alternative .NET programming languages and a few additional development utilities. After this point, the next several installments will dive headlong into the details of the C# programming language. Until then, as always, happy hacking.

Complete Source Control and Defect Management

Seapine Software™
changing the world
of software development

for Mac OS X



Effective source code control and defect tracking require powerful, flexible, and easy-to-use tools—Surround SCM and TestTrack Pro

- Complete source code control with private workspaces, automatic merging, role-based security, and more
- Comprehensive defect management — track bug reports and change requests, define workflow, customize fields
- Fast and secure remote access to your source files and defects — work from anywhere
- Advanced branching simplifies managing multiple versions of your products
- Link code changes with defects and change requests — know who changed what, when, and why
- Scalable and reliable cross-platform, client/server solutions support Mac OS X, Windows, Linux, and Solaris
- Exchange data using XML and ODBC, extend and automate with SOAP support
- Licenses priced to fit your budget

Seapine Software Product Lifecycle Management
Award winning, easy-to-use software development tools

Seapine
Surround SCM

Seapine
TestTrack PRO



**Download Surround SCM
and TestTrack Pro at**
www.seapine.com
or call 1-888-683-6456



all product names listed herein are registered trademarks of their respective owners. All rights reserved.

By Steve Gehrman

A simple debugging tool for Cocoa

Another weapon in the fight against bugs

Here's the problem: You write an application, test it, fix the bugs and everything works great while testing in-house. But, the moment you send it to the customer you get a ton of bug reports.

How could this be? You tested the code thoroughly, and you've been running the application for months throughout the development process. You're now getting bug reports you've never seen before. What happened?

The problem lies in the fact that it's just not possible for the developer or even a small team of in-house testers to uncover all the bugs no matter how hard they try. Software is just too complex, and there are just too many variations in machines, OS versions, and interactions with other installed software to catch everything.

So, what can you do? The only solution is to have your users help in the testing/debugging process. These users/testers are commonly referred to as beta testers. There's only one problem with this solution. Your beta testers aren't developers, and have no clue how the code works, so the bug reports you receive may not be easy to decipher or reproduce. What you really need is a fool proof way for a user to report a problem automatically with information in a form that helps the developer determine what went wrong.

In this article, I explain a simple technique for doing this. Using this technique, when a problem occurs in your application, a window will appear with information describing what happened, and a stack trace of where the problem occurred. There is also a button in the window to automatically email this information to the developer. That way, all the developer has to do is check their email, examine the reports, and fix the bugs. Sounds cool, right?

The key to this debugging tool is exceptions. Most Cocoa objects will throw an exception when passed a bad parameter or when something unexpected happens. Catching and displaying exception information is the key to this debugging aid. Catching exceptions can't find every type of software bug in your application, but it will uncover a surprising number of problems and uncover lots of trouble

spots in your code. I've been using this code in my own application, "Path Finder," for two years now, and it has proven indispensable.

The remainder of this article explores the code used to do this and explains how it works. All the code is downloadable in a simple example application. If you're like me and like to see the working code before reading the rest of the article, download the example application here from the MacTech website at www.mactech.com/src/19.12.

THE CODE:

Cocoa has two built in classes for exception handling: `NSException` and `NSExceptionHandler`. These two classes are located in the `ExceptionHandler.framework`. The first step is to add this framework to your application. I'm using XCode, so to do this I go to the "Project" menu and choose "Add Frameworks..." and navigate to `/System/Library/Frameworks/` and choose `ExceptionHandler.framework`.

Let's start by looking at the `NSExceptionHandler` class. `NSExceptionHandler` is really simple and does the work of telling us when an exception has occurred in our application. All we need to do is create a delegate object that will be sent a message when an exception occurs.

I created a class called `NTEExceptionHandlerDelegate` to do this. Here's the code, it's very short.

```
@implementation NTEExceptionHandlerDelegate
- (id)initWithEmail:(NSString*)emailAddress;
{
    self = [super init];
    _emailAddress = [emailAddress retain];

    [[NSExceptionHandler defaultExceptionHandler]
    setDelegate:self];
    [[NSExceptionHandler defaultExceptionHandler]
    setExceptionHandlingMask:NSLogAndHandleEveryExceptionMask];

    return self;
}

- (void)dealloc;
{
    [[NSExceptionHandler defaultExceptionHandler]
    setDelegate:nil];
}
```

Steve Gehrman is the founder and lead developer at CocoaTech. Feel free to contact Steve at steve@cocoatech.com.


```

setDelegate:nil];
[_emailAddress release];
[super dealloc];
}

// used to filter out some common harmless exceptions
- (BOOL)shouldDisplayException:(NSEException *)exception;
{
    NSString* name = [exception name];
    NSString* reason = [exception reason];

    if ([name isEqualToString:@"NSImageCacheException"] ||
        [name isEqualToString:@"GIFReadingException"] ||
        [name isEqualToString:@"NSRTTException"] ||
        [name
isEqualToString:@"NSInternalInconsistencyException"] &&
[reason hasPrefix:@"lockFocus"]])
    {
        return NO;
    }

    return YES;
}

- (BOOL)exceptionHandler:(NSEExceptionHandler *)sender
shouldLogException:(NSEException *)exception mask:(unsigned
int)aMask;
{
    // this controls whether the exception shows up in the console, just return YES
    return YES;
}

- (BOOL)exceptionHandler:(NSEExceptionHandler *)sender
shouldHandleException:(NSEException *)exception mask:(unsigned
int)aMask;
{
    // if the exception isn't filtered by shouldDisplayException, display the information
    to the user
    if ([self shouldDisplayException:exception])
        [[NTEExceptionHandlerController alloc]
initWithException:exception emailAddress:_emailAddress];

    return YES;
}

@end

```

For the init method of NTEExceptionHandlerDelegate, I pass it an email address. This is the email address where you want to receive the exception reports. You need to allocate one of these objects somewhere in your code. In the sample application, I allocate it in +[MyDocument initialize]. If you run the sample application, be sure to change the email address so you will receive the emailed reports.

Next, I call two methods using the shared default NSEExceptionHandler object:

```

[[NSEExceptionHandler defaultExceptionHandler]
setDelegate:self];
[[NSEExceptionHandler defaultExceptionHandler]
setExceptionHandlerMask: NSLogAndHandleEveryExceptionMask];

```

The call to setDelegate sets this instance of NTEExceptionHandlerDelegate as the NSEExceptionHandler's delegate. The second call to setExceptionHandlerMask tells NSEExceptionHandler which exceptions I'm interested in handling. I pass NSLogAndHandleEveryExceptionMask to tell it to send me every exception.

There were a few harmless exceptions that were happening frequently in Path Finder that I decided to filter out. That way my email box doesn't get full of emails that I've already determined to be OK. The exception filtering happens in the method shouldDisplayException. You may choose to remove or modify this code.

In order for the NTEExceptionHandlerDelegate instance to receive the exceptions, it must implement two methods found in the informal protocol located in NSEExceptionHandler.h.

```

@interface NSObject(NSEExceptionHandlerDelegate)
- (BOOL)exceptionHandler:(NSEExceptionHandler *)sender
shouldLogException:(NSEException *)exception mask:(unsigned
int)aMask;
- (BOOL)exceptionHandler:(NSEExceptionHandler *)sender
shouldHandleException:(NSEException *)exception mask:(unsigned
int)aMask;
@end

```

These are the messages sent by the NSEExceptionHandler to the delegate when an exception occurs. In the code above you can see that in the method exceptionHandler:shouldLogException:, I just return YES. This tells the NSEExceptionHandler to log the exception in the console. In the next method exceptionHandler:shouldHandleException:mask I call the code to display the exception to the user in a window. This is where the magic happens. This window is handled in by the NTEExceptionHandlerController class.

Let's look inside NTEExceptionHandlerController to see what it does.

```

- (id)initWithException:(NSEException *)exception
emailAddress:(NSString *)emailAddress;
{
    self = [super init];

    _displayFont = [[NSFont fontWithName:@"Monaco" size:10.0]
retain];
    _emailAddress = [emailAddress retain];

    // load the nib
    if (![NSBundle loadNibNamed:@"NTEExceptionHandler" owner:self])
    {
        NSLog(@"Failed to load NTEExceptionHandler.nib");
        NSBeep();
    }
    else
    {
        [self displayCrashReport:exception];
    }

    // center the window, show the window
    [window center];
    [window makeKeyAndOrderFront:nil];
}

return self;
}

```

The init call loads the nib containing the window, and if successful, calls [self displayCrashReport:exception]. The window loaded from the nib contains a single NSTextView. displayCrashReport's job is to fill that text view with the information extracted from the exception. NTEExceptionHandlerController has a simple helper method called displayText to write a string into the NSTextView.

EXTREME to the MACS!

Aria Extreme

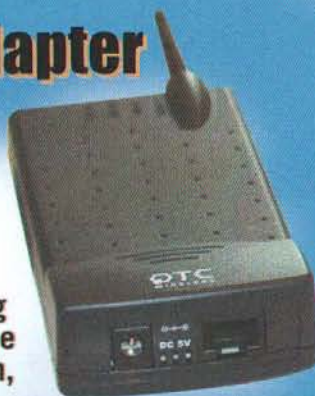
\$78.95



Add AirPort Extreme to any G3 or G4 PowerBook! Get blazing speeds up to 54 Mbps (five times AirPort Speed!), comes with an external stub antenna for better range than internal cards! (requires Mac OS X 10.2.6 or above)

**OTC Wireless
802.11g Ethernet Adapter**

\$164.95



Driverless Ethernet to 802.11g converter. Simply plug into the Ethernet port of any Macintosh, PC, or even printer to put it transparently on your 802.11g (54 Mbps) network! Use with a hub to enable multiple devices with one adapter!

DEV DEPOT®

www.devdepot.com

877-DEPOT-NOW

**Use AirPort
Extreme with
any Macintosh!**

AirPort Extreme (802.11g) PCI Card

\$98.95

Easy to install! Step up to
AirPort Extreme (54 Mbps)
for any PCI Macintosh!
(requires Mac OS X 10.2.6 or above)

AirPort Extreme

AirPort Extreme Omni Antenna 5 dBi

\$78.95

Want more range from your AirPort
Extreme base station? This 5 decibel
antenna is 30% more powerful than
other brands!

We also offer a full line of AirPort (802.11b) products!

**Mac Wireless
USB
to 802.11b
Converter**



**AirPort PCI
to 802.11b
Converter**



**AirPort 802.11b
PCMCIA Card**



**Belkin Wireless
Cable/DSL Gateway
Router**



\$78.95

\$128.95

\$88.95

\$112.95

Here's the code:

```
(void)displayCrashReport:(NSEException*)exception;
{
    NSMutableArray *args = [ NSMutableArray array ];
    NSString* appName = [self applicationName];
    NSString* stackTrace;

    // display instructions
    [self displayText:appName];
    [self displayText:[NSString stringWithFormat:@" has
encountered an unexpected error. Please email this report to
%@ so it can be fixed in the next release.", _emailAddress]];
    [self displayText:@"\r\n"];

    // display version
    [self displayText:appName];
    [self displayText:@" "];
    [self displayText:[NTUtilities applicationVersion]];
    [self displayText:@"\r\n"];

    // display OS version
    [self displayText:[NTUtilities OSVersion]];
    [self displayText:@"\r\n"];

    [self displayText:[exception name]];
    [self displayText:@"\r\n"];
    [self displayText:[exception reason]];
    [self displayText:@"\r\n"];

    stackTrace = [[exception userInfo]
objectForKey:NSStackTraceKey];
    if (stackTrace)
    {
        // add the process id
        [args addObject:@"-p"];
        [args addObject:[self applicationProcessID]
stringValue];

        // must add each stack trace as individual arguments
        NSScanner *scanner = [NSScanner
scannerWithString:stackTrace];
        NSString* output;

        NSMutableCharacterSet *whitespace = [NSMutableCharacterSet
whitespaceCharacterSet];

        while (YES)
        {
            if ([scanner
scanUpToCharactersFromSet:whitespace intoString:&output])
                [args addObject:output];
            else
                break;
        }

        _atosTask = [[NTTaskController alloc]
initWithDelegate:self];
        [_atosTask runTask:NO toolPath:@"usr/bin/atos"
directory:@"/" withArgs:args];
    }
}
```

We start by displaying the application name and instructions to the user telling them what has occurred and what to do next. Next we display the version of the application, the version of the OS, the exception name, and the exception reason. Next is the stack trace. Inside the exception userInfo dictionary is an NSString with the key NSTStack TraceKey. This string looks something like this:

```
0x8c7a1f04 0x97e54804 0x97df1820 0x01e58aa0 0x930f9cac 0x9311a3cc
0x9315c54c 0x930f36b8 0x9315c168 0x9312de6c 0x930c102c 0x930a8e20
0x930b1dac 0x9315fc58 0x00117f74 0x000038e0 0x00003760 0x00001000
```

That isn't too useful. What we really need is for those numeric addresses to be converted into symbols. Fortunately, there is a built in unix command line tool that does this called "atos". Open the Terminal and type "man atos" for more information.

In order to run "atos" we first need convert the string of addresses in to an array of individual address strings. This is used as the parameter to "atos". atos takes this array of addresses and converts each one to a human readable symbol. In Cocoa this is easily accomplished using an NSTask. In this code I'm using a wrapper around NSTask called NSTaskController which makes running an NSTask even easier. Basically, it runs the task and sends us the output and takes care of all the details automatically. Next, the code takes the output and adds it to the windows NSTextView. Look at the documentation on NSTask and look at the code of NSTaskController to see how this works. It's very straightforward.

Here's some sample output from atos. Given the input string of: "0x8c7a1f04 0x97e54804 0x97df1820 0x01e58aa0 0x930f9cac 0x9311a3cc 0x9315c54c 0x930f36b8 0x9315c168 0x9312de6c 0x930c102c 0x930a8e20 0x930b1dac 0x9315fc58 0x00117f74 0x000038e0 0x00003760 0x00001000", we get this:

```
_NSEExceptionHandlerExceptionRaiser (in ExceptionHandling)
+[NSEException raise:format:] (in Foundation)
-[NSCFArray objectAtIndex:] (in Foundation)
-[MyDocument exceptionAction:] (in MyDocument.ob)
(MyDocument.m:72)
-[NSApplication sendAction:to:from:] (in AppKit)
-[NSControl sendAction:to:] (in AppKit)
-[NSCell _sendActionFrom:] (in AppKit)
-[NSCell trackMouse:inRect:ofView:untilMouseUp:] (in AppKit)
-[NSButtonCell trackMouse:inRect:ofView:untilMouseUp:] (in
AppKit)
-[NSControl mouseDown:] (in AppKit)
-[NSWindow sendEvent:] (in AppKit)
-[NSApplication sendEvent:] (in AppKit)
-[NSApplication run] (in AppKit)
_NSApplicationMain (in AppKit)
_main (in main.ob) (main.m:13)
start (in ExceptionHandlerExample)
start (in ExceptionHandlerExample)
0x00001000 (in ExceptionHandlerExample)
```

That's a whole lot easier to understand than the string of addresses.

So, we are almost done. We have a window which contains all the information on the exception and a stack trace telling us exactly where the exception occurred. Now all we need to do is email this information to the developer.

Cocoa has a very simple interface for sending an email using the class NSMailDelivery. NSMailDelivery is part of the Message.framework, so you will need to add the Message.framework to your project. I'm using XCode, so to do this I go to the "Project" menu and choose "Add Frameworks..." and navigate to /System/Library/Frameworks/ and choose Message.framework.

```
(void)emailAction:(id)sender;
{
    BOOL mailSent=NO;
```



```

NSString *subject = [NSString stringWithFormat:@"%s@%s", [self applicationName]];
exception report", [self applicationName]];

mailSent = [NSMIMEDelivery deliverMessage:[textView
string] subject:subject to:_emailAddress];

if (!mailSent)
    NSBeep(); // you may want to handle the error

[window close];
}

```

The email button in the window is set to send the action emailAction. In email action I construct a string for the message subject and use the contents of the NSTextView as the email body. Sending the email requires just one line of code:

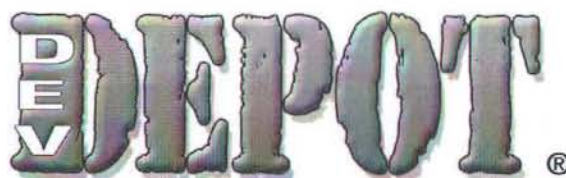
```

mailSent = [NSMIMEDelivery deliverMessage:[textView string]
subject:subject to:_emailAddress];

```

SUMMARY:

So, that's all there is to it. Add this to your application, send it out to beta testers and get back accurate reports of trouble spots in your code. This debugging tool doesn't catch all types of bugs. There are many types programming errors that don't throw exceptions and therefore won't be detected by this system, but it is one more powerful tool in your arsenal against bugs.



www.devdepot.com

**Don't miss us at Macworld!
DevDepot Has It All!**

**DevDepot sells the tools, toys and
technology to put more muscle into
your Mac!**

**Visit the Official Store, managed by
DevDepot, for show specials and
great new products from all over
the show floor.**

ListSTAR®

www.liststar.com

The most flexible email processing system available. Easily create mailing lists or email-on-demand services. Use built in rules and/or AppleScript/AppleEvents to handle any email task, no matter how simple or complex. Demo available.

MacRADIUS™

www.macradius.com

The easiest to use RADIUS server available. The groups feature allows you to make changes for a large number of users in one easy step. Enabling or disabling access for a user or a group of users is a one click operation, without having to stop the server. Support for AppleScript/AppleEvents makes it easy to control MacRADIUS. Demo available.

SimpleText Filter

Plug-in for EIMS*

www.mcfsoftware.com/stf/

Easy to use header and content filtering. Scan incoming messages for sequences of text, digits, etc. Base64 messages are decoded so you can check for content despite attempts to hide the text. Demo available.

Auto Reply

Plug-in for EIMS*

www.mcfsoftware.com/ar/

This plug-in allows you to easily set up auto-reply messages for users. Addresses that have been sent auto-reply messages are tracked, preventing auto-reply message loops. Demo available.

Address List Sorter

www.mcfsoftware.com/als/

Fast and powerful utility for sorting and cleaning email lists. Sort email address lists in alphabetical or domain order, remove duplicates and improperly formed addresses. Demo available.

*EIMS—Eudora Internet Mail Server (www.eudora.co.nz)



**...simply
dependably
engineered**

www.mcfsoftware.com

By Steve Klingsporn

Writing Cocoa Applications in Java

Mac OS X is a great platform for software development. The combination of the Mac's elegant user experience, signature design and media tools, breakthroughs in mobile computing and UNIX-based underpinnings continues to lure developers away from other platforms. Apple has done a great deal of performance tuning in version 10.3 "Panther," optimizing the Carbon and Cocoa application frameworks while upgrading the underlying Mach kernel and BSD environment. Developers have a diverse choice of programming languages, from traditional assembly, C, and C++ to more dynamic languages like Objective-C, Java, JavaScript and Python. Apple ships Panther with "Xcode," its integrated development environment, which includes comprehensive documentation, sample code and the usual compilers and command-line tools. Xcode is an evolution of Project Builder, and adds neat features like code completion, fix and continue, predictive compilation, zero-link, distributed builds and a graphical debugger. Developers can choose between graphical tools like Xcode and Metrowerks CodeWarrior or more traditional command-line text editors and utilities. Apple rounds off its UNIX offerings with an X11 environment, which makes it extremely attractive to developers coming from Linux and FreeBSD camps. Apple is now the top vendor of high-volume UNIX systems, and Mac users can *finally* enjoy a truly remarkable, stable, and secure operating system.

Java has surpassed C and C++ as the world's most popular programming language, and is standard fare at most colleges and universities. Sun describes Java as "simple, object-oriented, distributed, interpreted, robust, secure, architecture neutral, portable, high-performance, multithreaded, and dynamic." Developers compile Java source into platform-independent byte code that executes identically across host platforms in a virtual machine (VM) environment. Sun's "Hotspot" VM dynamically compiles this byte code into native object code at runtime, and its performance often equals or exceeds that of compiled C and C++. Apple realized this when moving its WebObjects

application server to Java, as did Netscape when it profiled the performance of its Java ("Rhino") and C-based ("SeaMonkey") JavaScript engines. The performance problems that plagued Java in the early days have been addressed, and in most cases, the only noticeable tradeoffs are slightly longer application launch times and the memory overhead incurred by the VM. This environment offers an additional level of stability and security that C and C++ can't match, protecting applications from crashing and unauthorized resource access through the use of exceptions and a policy-based security model. Java has taken off in the server space due to many of these advantages, where stability and security are critical. Server-side Java development is pure joy on Mac OS X, as you can run any of the popular J2EE-compliant application servers, most popular relational databases, and you can easily deploy on Mac OS X Server or any other Enterprise-class UNIX servers while developing on your PowerBook and listening to iTunes.

Apple ships Java 2 Standard Edition (J2SE) versions 1.3.1 and 1.4.1 with Panther, and these releases are available to developers still using Jaguar via the Software Update service. Apple includes a number of notable improvements and innovations in its J2SE implementations, including the sharing of commonly used classes across VM instances, and using the Cocoa framework as the basis for its AWT and Java2D implementations. The net result is arguably the best and most tightly integrated implementation of Java 2 available on any platform. In addition to the standard tools from Sun, Apple includes a "Jar Bundler" utility that assists in generating double-clickable applications from JAR files, and "JavaBrowser," a quick and useful class browser that allows the browsing of Java package hierarchies and their associated class definitions, source code and documentation without switching back and forth between Xcode and multiple pages in a web browser. These two applications live in `/Developer/Applications/Java Tools`, and are installed by the Xcode installer. **Figure 1** depicts JavaBrowser displaying the source code for the `java.util.StringTokenizer` platform class.

Steve Klingsporn is an independent software developer living in Chicago with his cat, Sonya. He has an 11 year history of working at companies such as Apple, Netscape, and Sun. He is available for independent Mac OS X, Java and web development, and can be reached at steve@buzzlabs.com.

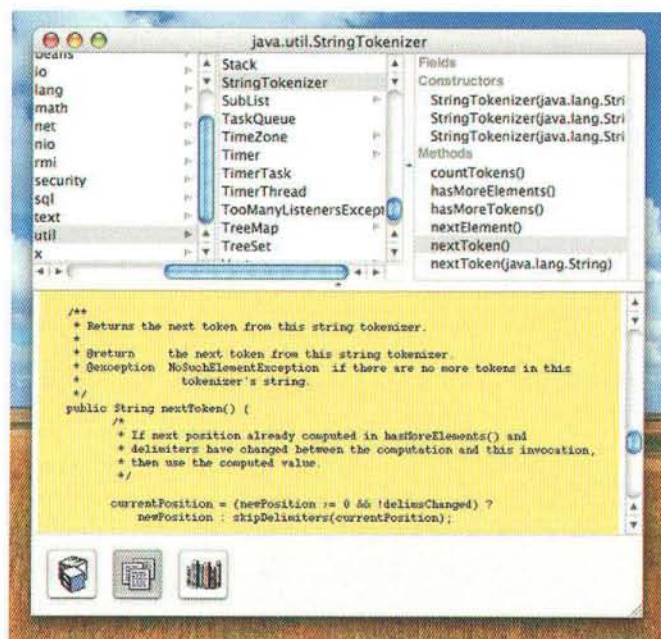


Figure 1: JavaBrowser displaying the source code to the StringTokenizer class.

COCOA JAVA

Cocoa is the evolution of the object-oriented application frameworks developed in the early 90's at NeXT, and was instrumental in the transition to Mac OS X at Apple. Developers who are interested in writing new applications for the Mac will probably choose Cocoa over Carbon because of its modern, object-oriented nature and award-winning rapid user interface design tool, "Interface Builder." With Interface Builder, you visually construct your application's user interface, dragging, dropping, configuring and tying components to "outlets" and "actions" which map directly to properties and methods implemented in your source code. Interface Builder generates "NIB files" containing frozen objects, which are packaged in directories that correspond to the various human languages your application is localized into. Organizing NIB files this way together with localized program strings gives developers a simple and powerful mechanism for providing their applications in many different languages and markets simultaneously.

Cocoa is written in the Objective-C language, which consists of a small number of extensions to ANSI C that enable object-oriented programming and dynamic runtime features. Objective-C is a more elegant and concise language than C++, lacks many of its drawbacks and pitfalls, and has a unique syntax for defining classes, categories, protocols and message passing between objects. Despite the power and simplicity of this language, C++ won the language war in the mid 90's, and Java's popular syntax more closely

data
There are Users
data
and Losers...

Which
would you rather
be

Introducing

Data Safety System

Backup, Undelete and Recover files.



Data Users get it!

www.prosofteng.com

PROSOFT
engineering, inc.

These products are only available for the Mac OS, so your Windows friends will still be losers...

©2003 Prosoft Engineering, Inc., All rights reserved

resembles C++, while borrowing many of Objective-C's features. James Gosling, Java's mastermind, credits Objective-C for inspiring many of the language's dynamic features, and today uses his PowerBook for most of his research and development work. While Objective-C may be the language that Apple would like to see developers using, it acknowledges Java's popularity and utilizes the "Java Bridge" technology the WebObjects team developed to facilitate building fully native Cocoa applications in pure Java. While you can continue to develop platform-independent Java applications using "Swing" and AWT, only Cocoa offers a true Aqua user experience and access to platform-specific features that Mac users have come to expect. Cocoa Java is a very attractive option for new and existing developers, leveraging the best aspects of these technologies without requiring the additional learning curve of mastering an unfamiliar programming language. You can freely mix Objective-C and Java code in your applications, and can design your classes so Cocoa-dependent code is separate from more generic code, which can continue to run unmodified on other host platforms.

CREATING YOUR PROJECT

To create a new Cocoa Java project, launch Xcode (in /Developer/Applications), and select "New Project..." from the "File" menu. Select "Cocoa Java Application," click "Next," and tell Xcode the name and location of your new project. If you are planning on developing a document-based application utilizing the `NSDocument` class, you may want to select "Cocoa Java Document-based Application" instead.

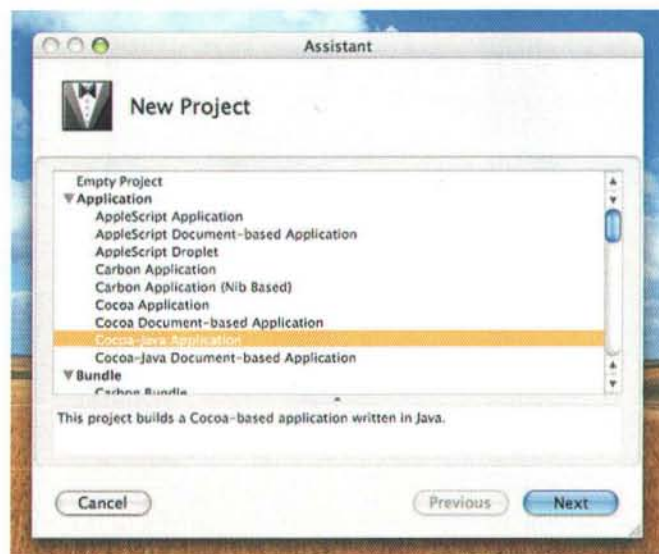


Figure 2: Using Xcode to create a new Cocoa Java project.

Xcode will create a new project directory for you based on its template for a Cocoa Java application, and will put

some placeholder files into this directory, including a "MainMenu.nib" file for editing in Interface Builder, links to the AppKit, Cocoa and Foundation frameworks, and a "main.m" Objective-C stub file that is called only to launch your application, and is actually optional. **Figure 3** shows what a new Cocoa Java project window should look like. The name of your project, of course, will be different from the one used in this article. If you are using Project Builder on Jaguar, things will be arranged differently. A considerable number of articles have been written about Cocoa development in Project Builder.

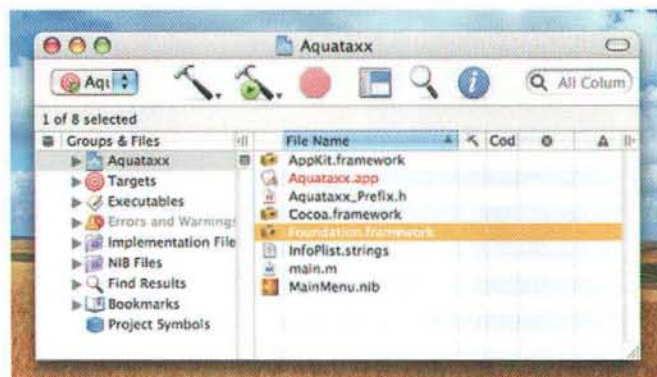


Figure 3: A new Cocoa Java application project window.

At this point, you will begin to follow the usual iterative Cocoa development patterns. Much has been written in previous articles and tutorials about this process, which generally consists of using Interface Builder to design classes in the "Classes" tab of your NIB file window, specifying instances of these classes in the "Instances" tab by dragging or manually creating them, and drawing out the connections between them. You then either auto-generate or edit your source code files, implementing the various member variables and methods associated with the outlets and actions you define in the "Classes" tab. You should keep all of your program strings in a text file called "Localizable.strings," and use `NSBundle.localizedString` to retrieve them instead of hard-coding them in your source. You can generate different language bundles for other localizations containing this file and any NIB files that have localization-specific changes. "MainMenu.nib" is the default NIB file that Xcode generates from its Cocoa Java project template, and should be double-clicked to launch Interface Builder when you are ready to begin specifying the menus and windows for your application. You can break your application's user interface up into multiple NIB files to decrease your application's launch time and provide your localizers with files that are easier to manage and edit.

DIFFERENCES FROM OBJECTIVE-C

The obvious difference from programming in Objective-C is that you will be using Java to implement your source files.

The Cocoa AppKit and Foundation Kit frameworks are implemented in the `com.apple.cocoa.application` and `com.apple.cocoa.foundation` packages. You should import these packages in full, or at least the classes from them you plan on using, in each of your source files that depends on Cocoa. You are simply using Cocoa for your user interface and platform integration instead of AWT or JFC ("Swing"). Everything else more or less remains familiar, and you can use any of the tried and true Java platform classes you like mixed in with your use of Cocoa. You can add the paths to your class, source and documentation files for your application to JavaBrowser if you wish, and conveniently browse them in conjunction with the Java and Cocoa platform classes. You can even use "Ant" to build portions or the entirety of your project, though this will require you to manually write a "build.xml" file and do a bit of extra work. You will find that most of the Cocoa sample code on the web and most of the mailing lists, books and other resources listed at the end of this article are written in Objective-C, so a rudimentary understanding of the language and its message passing syntax will prove useful. Converting between Objective-C and Java will become second-nature as soon as you get the hang of its syntax peculiarities, and you will begin to appreciate the conventions Apple took in converting the API to Java, often using familiar interfaces instead of protocols and the like. In most cases, there is no penalty (or "toll") for subclassing or calling back and forth between the two languages across the Java Bridge, and Apple's documentation describes cases in which you should pay special attention to performance or memory management issues.

One key difference between Objective-C and Java Cocoa is that Apple has not provided Java equivalents for many of the I/O, string manipulation and other such classes in the Foundation Kit, including `NSFileHandle`, `NSThread`, `NSValue` and `NSScanner`, expecting developers to instead use standard Java platform counterparts like `java.io.File`, `java.lang.Thread`, `java.lang.Object` and `java.util.StringTokenizer`. Use `java.lang.String` instead of `NSString`; `NSAttributedString` and other Cocoa classes that take strings accept them. Java's network and file I/O classes are not only familiar to seasoned Java developers, but often offer more elegant solutions than their Objective-C counterparts. Java's platform advantages really start to become self-evident when using features like reflection, serialization, and remote method invocation (RMI), not to mention libraries like JDBC, which enables a Cocoa developer to easily write a vendor-neutral relational database application with a native Cocoa user experience in an afternoon. Any of the countless Java packages and libraries available can be leveraged by your Cocoa Java applications, saving you time and allowing you to concentrate on implementing application-specific functionality. This truly powerful combination of technologies should not be overlooked by anyone considering using Objective-C for their next killer Cocoa app.

New

PrimeBase 4.2 Replication Server

Check out the fully programmable Replication Server

- Bidirectional Updates supported
- Update 3rd-party DBMS
- Send Emails
- Post/get Data to/from Websites

**SQL-Runtime Plugin
for REALbasic**
\$ 499,-

All PrimeBase Server Software

- SQL-Runtime Libraries available
- SQL Database Server
- Application Server
- Replication Server
- Open Server

Available on the most popular platforms

- Completely cross-platform
- Full-text searching and indexing
- Mac OS & OS X
- Linux
- Solaris
- IBM AIX
- all Windows platforms

 **PRIMEBASE**

SNAP Innovation GmbH
Altonaer Poststraße 9a
D-22767 Hamburg / Germany

www.primebase.com

e-mail: info@primebase.com

Fon: ++49 (40) 389 044-0

Fax: ++49 (40) 389 044-44

MIXING JAVA AND COCOA

For the purposes of this article, we will be examining some of the code from "Aquataxx," a Cocoa Java implementation of the classic arcade strategy game, "Ataxx," shown in **Figure 4**. Aquataxx is a comprehensive example of using Cocoa features like brushed metal windows, drawers, sheets, bouncing Dock icons, text editing, tab views, user defaults, localized strings, drag and drop, sound, animation and drawing in custom `NSView` subclasses. In addition, it sports a game engine with impressive AI, networked game play and messaging, a standalone network "roster server" for finding other online players, and some advanced threading techniques, all written in portable, platform-agnostic Java. The result is a portable implementation of Ataxx with a challenging computer player, a rich, themable Cocoa user experience, network play and chat, and more. Writing the same game in Objective-C would have taken longer than the two weeks it took to write Aquataxx.

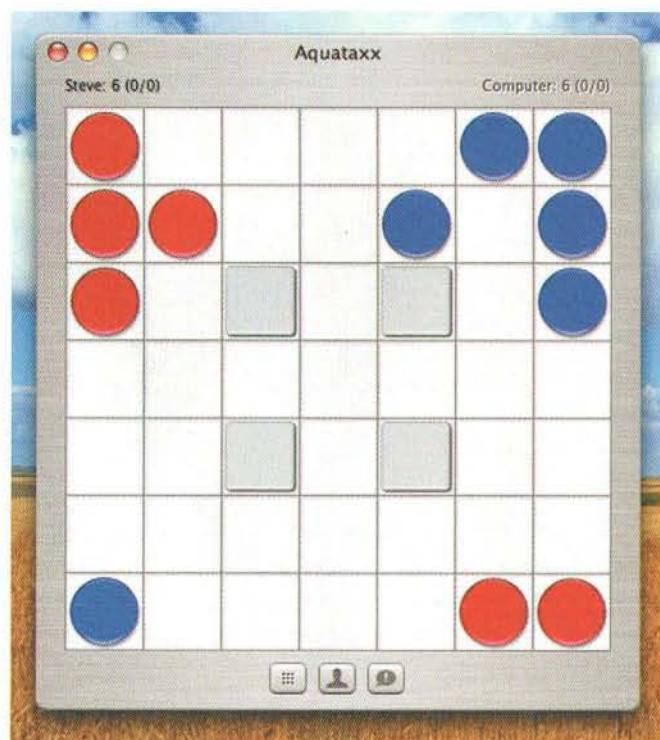


Figure 4: *Aquataxx, a Cocoa Java game used as an example in this article.*

Listing 1 depicts using Cocoa and Java platform classes together to implement the action method that is invoked by the "About Aquataxx" item in the "Aquataxx" application menu. Note that the method is public and a regular `java.lang.Object` is used for the sender parameter. We splice the Java version from `java.lang.System` into the `version` `NSString` and display the game's about sheet.

Listing 1:

A simple Cocoa "action" method, linked from the NIB file in Interface Builder.

```
/**
 * Shows the about sheet
 * @param sender the sender of the message
 */
public void showAboutSheet(Object sender)
{
    /* Set the version string, if need be */
    NSString versionField =
        (NSString)mAboutPanel.contentView().
        viewWithTag(1);

    String versionString = versionField.stringValue();
    if (versionString.indexOf("{0}") > -1)
    {
        StringBuffer buffy = new
            StringBuffer(AtaxxApplication.APPLICATION_VERSION);
        buffy.append(" - Java ");
        buffy.append(System.getProperty("java.version"));
        versionString = MessageFormat.format(versionString,
            new Object[] { buffy.toString() });
        versionField.setStringValue(versionString);
    }

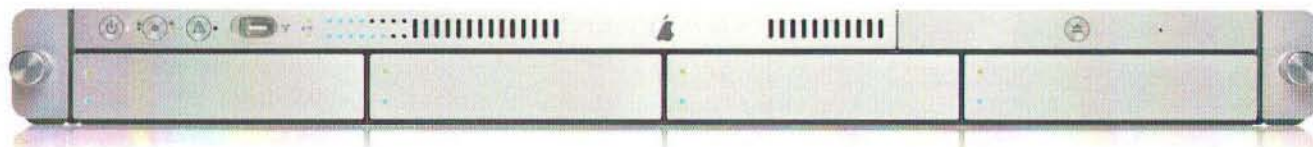
    /* Show the sheet */
    NSApplication.sharedApplication().beginSheet(
        mAboutPanel, mGameWindow, this, null, null);
}
```

MEMORY MANAGEMENT AND THREAD SAFETY

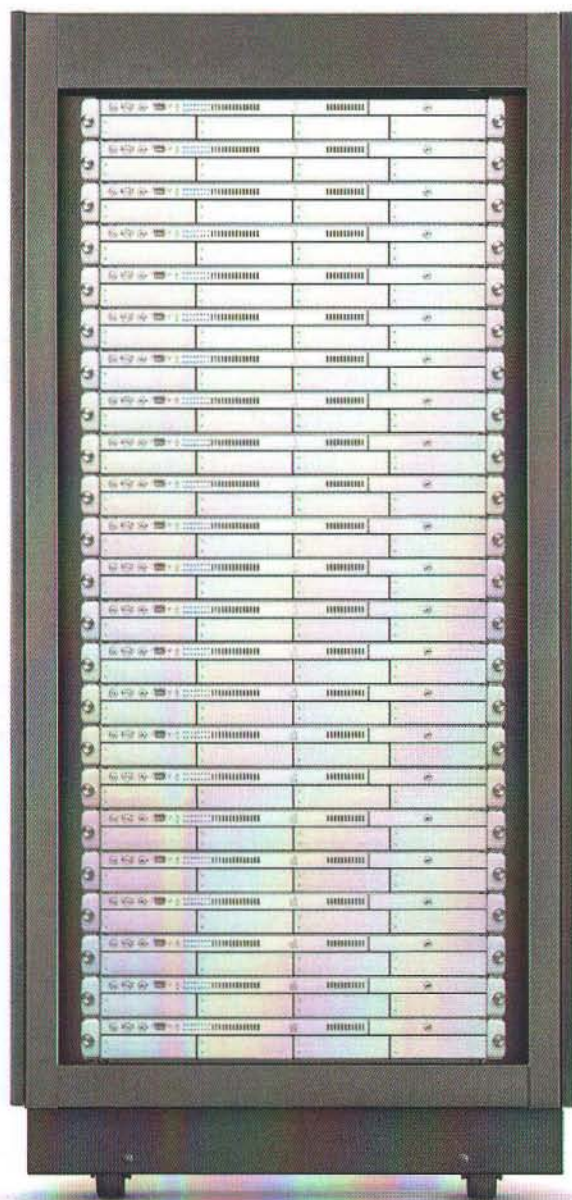
In most cases, you will not have to explicitly worry about memory management in your Cocoa Java applications. The usual warnings regarding hanging onto references and what not still apply, and Apple has a document that is referenced at the end of this article that explains the exceptions to these general rules. In some cases, to get around complicated interactions between Cocoa and Java objects, you will need to wrap objects in an `NSArray`; in others, you may have to catch possible `NSExceptions` that can be thrown by Cocoa object constructors. For the most part, the tricky `retain` and `release` memory management conventions used in Objective-C can be avoided; another one of Java's advantages improves the Cocoa development experience!

Java developers are used to satisfying the needs of most modern applications, especially those that use network I/O, with multiple threads. Java encourages the use of multithreading, and Java threads (`java.lang.Thread`) on Mac OS X are implemented as native Mach threads, so they have excellent performance. One of the inherent pitfalls of multithreaded programming is *thread safety*, and many aspects of the Cocoa frameworks are not yet guaranteed to be thread safe, meaning you cannot safely call many aspects of Cocoa, including your user interface, from background threads such as one that blocks on incoming network I/O. Since network socket I/O blocks in J2SE 1.3.1, and holding up the *main thread* in which many aspects of Cocoa including your user interface run in can result in unacceptable performance, a mechanism is needed to enable background threads to tell the main thread to execute code that interacts with Cocoa. `NSObject` in Objective-C has a method,

Xserve



Density optimized rack mounted Mac OS X Server



UNIX-based Server Solutions from Apple

Nearly half a terabyte of storage per 1U

630 gigaflops of processing power

Hot-Swappable drives

Industry-standard 1U rack-optimized design

There has never been a better time to buy...

Small Dog Electronics carries
factory refurbished models too
offered at substantial savings!

(subject to availability)



**Small Dog
Electronics**
www.smalldog.com

1673 MAIN STREET, ROUTE 100, WAITSFIELD, VERMONT



Apple Specialist

To learn more: <http://www.smalldog.com/xserve/>

performSelectorInMainThread, but unfortunately this functionality is not available in Cocoa Java. To get around this, the most popular strategy is to subclass `NSApplication` and implement `sendEvent` to handle application-defined events that correspond to blocks of code that need to be called in the main thread. Code that is called from your `sendEvent` implementation is guaranteed run in the main thread. Failing to observe the fact that many aspects of Cocoa are not thread safe will result in unexpected, random crashes (not exceptions) that can be hard to reproduce and debug and appear to be in Cocoa's Objective-C code. These types of crash logs can be confusing to developers programming in pure Java, yet can often help in tracking down sections of code that should be run in the main thread. **Listing 2** is the full source for `AtaxxApplication`, the custom `NSApplication` subclass used in `Aquataxx` to handle events posted by background threads. **Listing 3** depicts an example of posting such an event. The full source for `Aquataxx` is referenced at the end of this article, and can be examined for further examples of using this technique to assure your application plays by proper threading rules.

Listing 2:

`AtaxxApplication`, an `NSApplication` subclass that handles application-defined events.

```
/* AtaxxApplication.java */

import com.apple.cocoa.foundation.*;
import com.apple.cocoa.application.*;

/**
 * A NSApplication subclass to add the ability to respond to application-defined events.
 */
public class AtaxxApplication
    extends NSApplication
{
    /** Application version string */
    public static final String APPLICATION_VERSION = "1.8";

    /** Application-defined event types */
    protected static final short NEW_MATCH_EVENT = 0;
    protected static final short SCROLL_CONVERSATION_EVENT = 1;
    protected static final short CONFIRM_CONNECTION_EVENT = 2;
    protected static final short UPDATE_ROSTER_EVENT = 3;
    protected static final short INCOMING_DROPPED_EVENT = 4;

    /**
     * Overridden to respond to custom events.
     * @param event the event being processed.
     */
    public void sendEvent(NSEvent event)
    {
        /* Only worry about application-defined events */
        if (event.type() == NSEvent.ApplicationDefined)
        {
            switch (event.subtype())
            {
                /* Create a new match */
                case NEW_MATCH_EVENT:
                {
                    AtaxxController.sharedController().
                        startNewMatch(event.data1(),
                                    event.data2());
                    break;
                }

                /* Scroll the conversation view */
                case SCROLL_CONVERSATION_EVENT:
                {
                    AtaxxController.sharedController().
                        scrollConversationView();
                }
            }
        }
    }
}
```

```
        break;
    }

    /* Confirm a connection request */
    case CONFIRM_CONNECTION_EVENT:
    {
        AtaxxController.sharedController().
            confirmConnectionRequest();
        break;
    }

    /* Update roster data source */
    case UPDATE_ROSTER_EVENT:
    {
        AtaxxController.sharedController().
            updateRosterDataSource();
        break;
    }

    /* Close the connection request sheet */
    case INCOMING_DROPPED_EVENT:
    {
        AtaxxController.sharedController().
            closeAnySheet(null);
        break;
    }
}
else
{
    super.sendEvent(event);
}
}
```

Listing 3:

A wrapper method that posts an application-defined event.

```
/**
 * Posts an application-defined event to the current application.
 * @param code the event code
 * @param data1 the first parameter
 * @param data2 the second parameter
 * @param front post to the front of the queue?
 */
public void postApplicationDefinedEvent(short code,
                                         int data1,
                                         int data2,
                                         boolean front)
{
    NSEvent event =
        NSEvent.otherEvent(NSEvent.ApplicationDefined,
                           new NSPoint(0, 0), 0, System.currentTimeMillis() /
                           1000.0, 0,
                           null, code, data1, data2);
    NSApplication.sharedApplication().postEvent(event, front);
}
```

IMPLEMENTING CUSTOM VIEWS

One of the more rewarding aspects of Cocoa programming is implementing your own `NSView` subclasses. `NSView` is the base class in the Cocoa user interface class hierarchy, and all of the user interface components you see and interact with inevitably inherit from it. If one of Apple's standard views does not suit the type of data or interaction you are trying to represent, as is the case with the game board and player score views in `Aquataxx`, you should implement a custom `NSView` subclass. You subclass `NSView` in the "Classes" tab of your NIB file window in Interface Builder, and create a corresponding Java source file just as you would if you were using Objective-

C. **Listing 4** is the complete source code for `AtaxxScoreView`, a custom view that draws both players' names and scores and highlights the active player with an etched appearance that looks similar to iTunes. For a more comprehensive and exciting example of a custom view that handles events and uses optimized drawing and drag and drop, take a look at the source for `AtaxxView`, which draws the game board and is found in the full source code distribution. Cocoa user interface programming is a lot more fun, intuitive, and rewarding than using "Swing!"

Listing 4:

The complete source for `AtaxxScoreView`, a custom `NSView` subclass.

```
/* AtaxxScoreView.java */

import com.apple.cocoa.foundation.*;
import com.apple.cocoa.application.*;

/**
 * A custom NSView subclass that displays an etched
 * metallic scoreboard for two players, and highlights
 * the player whose turn it is.
 */
public class AtaxxScoreView
    extends NSView
{
    /* The colors for drawing the scoreboard */
    public static final NSColor FOREGROUND_COLOR =
        NSColor.colorWithCalibratedRGB(.20f, .20f, .20f, 1.0f);
    public static final NSColor BACKGROUND_COLOR =
        NSColor.whiteColor();
    public static final NSColor FOREGROUND_UP_COLOR =
        NSColor.blackColor();
    public static final NSColor BACKGROUND_UP_COLOR =
        NSColor.grayColor();

    /* The (four) attributed score strings ((back, front) * 2) */
    private NSMutableAttributedString mScoreStrings[] = null;

    /**
     * Constructor
     * @param frame the frame rectangle
     */
    public AtaxxScoreView(NSRect frame)
    {
        super(frame);
        mScoreStrings = new NSMutableAttributedString[4];
    }

    /**
     * Tells the window server that we are not opaque (we are transparent).
     * @returns false, as we are transparent.
     */
    public boolean isOpaque()
    {
        return false;
    }

    /**
     * Draws the view
     * @param rect the update rectangle
     */
    public void drawRect(NSRect rect)
    {
        if (mScoreStrings != null)
        {
            NSMutableRect stencil = new NSMutableRect(bounds());
            stencil.insetRect(0.5f, 1.0f);
            stencil.setOrigin(new NSPoint(1.0f, 0.0f));
            NSGraphics.drawAttributedString(mScoreStrings[0],
            stencil);
            NSGraphics.drawAttributedString(mScoreStrings[2],
            stencil);
            stencil.setOrigin(new NSPoint(0.0f, 1.0f));
            NSGraphics.drawAttributedString(mScoreStrings[1],
            stencil);
            NSGraphics.drawAttributedString(mScoreStrings[3],
            stencil);
        }
    }
}
```

SOFTWARE LOCALIZATION MADE

easy

POWERGLOT FEATURE HIGHLIGHTS

- LOCALIZE CLASSIC, CARBON™, COCOA® AND PALM OS™ APPS
- LEVERAGE EXISTING TRANSLATIONS
- AUTOMATE WITH APPLESCRIPT®
- IMPORT /EXPORT TRANSLATION MEMORIES

www.powerglot.com
browse, translate, click, done.

PowerGlot Software • Localization tools for Mac® OS
www.powerglot.com
info@powerglot.com

Mac OS, Carbon, Cocoa and AppleScript are trademarks of Apple Computer, Inc.
Palm OS is a trademark of Palm, Inc.


```

stencil);
    }
}

/**
 * Creates the 4 attributed score strings, based on
 * the current score strings and player that is up.
 * @param player1 the string for the first player
 * @param player2 the string for the second player
 * @param up the player that is up (highlighted)
 * (0 = none, 1 = player 1, 2 = player 2)
 */
public synchronized void update(String player1,
                                String player2,
                                int up)
{
    /* For convenience in looping through the strings */
    String strings[] = { player1, player1, player2, player2 };
    NSMutableAttributedString newStrings[] =
        new NSMutableAttributedString[4];

    for (int i = 0; i < 4; i += 2)
    {
        /* Create the pair of attributed strings (foreground and background) */
        newStrings[i] =
            new NSMutableAttributedString(strings[i]);
        newStrings[i + 1] =
            new NSMutableAttributedString(strings[i + 1]);

        /* Create a range for the string pair */
        NSRange range = new NSRange(0, strings[i].length());

        /* Set the text alignment */
        NSMutableParagraphStyle paragraphStyle =
            new NSMutableParagraphStyle();
        paragraphStyle.setAlignment((i < 2) ?
            NSText.LeftTextAlignment :
            NSText.RightTextAlignment);
        newStrings[i].addAttributeInRange(
            NSAttributedString.ParagraphStyleAttributeName,
            paragraphStyle, range);
        newStrings[i + 1].addAttributeInRange(
            NSAttributedString.ParagraphStyleAttributeName,
            paragraphStyle, range);

        /* Set the font to the system font, size 10 (localization?) */
        NSFont font = NSFont.systemFontOfSize(10);
        newStrings[i].addAttributeInRange(
            NSAttributedString.FontAttributeName,
            font, range);
        newStrings[i + 1].addAttributeInRange(
            NSAttributedString.FontAttributeName,
            font, range);

        /* Set the foreground and background colors */
        NSColor foregroundColor = FOREGROUND_COLOR;
        NSColor backgroundColor = BACKGROUND_COLOR;
        if ((i < 2 && up == 1) ||
            (i > 1 && up == 2))
        {
            foregroundColor = FOREGROUND_UP_COLOR;
            backgroundColor = BACKGROUND_UP_COLOR;
        }
        newStrings[i].addAttributeInRange(
            NSAttributedString.ForegroundColorAttributeName,
            backgroundColor, range);
        newStrings[i + 1].addAttributeInRange(
            NSAttributedString.ForegroundColorAttributeName,
            foregroundColor, range);

        /* Fix up the attributes, whatever that does */
        newStrings[i].fixAttributesInRange(range);
        newStrings[i + 1].fixAttributesInRange(range);

        /* Set the actual attributed score strings */
        mScoreStrings[i] = newStrings[i];
        mScoreStrings[i + 1] = newStrings[i + 1];
    }

    /* Redraw the view! */
    display();
}

```

CONCLUSION

Java is an excellent choice for developing Cocoa applications. If you are a Java programmer, are new to Cocoa, or are not interested in learning Objective-C, you should give Cocoa Java a try. The popularity of the language and the sheer quantity of third-party libraries and tools provide Java programmers with an impressive array of pre-engineered solutions over those available in Objective-C. Apple continues to provide Java interfaces to the new features it adds to Cocoa. It's time for developers to take better advantage of this technology and provide Apple with the feedback they need to make Cocoa Java even better. The end result is more great applications for Mac OS X.

RESOURCES

The following resources will help you get started in learning about Cocoa Java programming. Sun has a general Java language tutorial, Apple has a Cocoa Java tutorial, and there are several sample applications installed with Xcode that will prove to be good references. The Aquataxx game distribution and full source code referenced in this article is available, and Apple and The Omni Group have good Cocoa development mailing lists that can be browsed and searched at <http://cocoa.mamasam.com>. If you have questions that you cannot find the answers to, or would like to discuss Cocoa Java programming in general, you can contact the author at steve@buzzlabs.com.

Aquataxx game distribution and Cocoa Java source code
<http://buzzlabs.com/aquataxx/>

The Java Tutorial
<http://java.sun.com/docs/books/tutorial/>

Developing Cocoa Java Applications: A Tutorial
<http://developer.apple.com/documentation/Cocoa/Conceptual/JavaTutorial/>

Apple Cocoa Java Examples (Xcode): BlastApp, Sketch, SimpleToolbar and TextEdit
<file:///Developer/Examples/Java/AppKit/>

Java/Objective-C Language Integration: Java Memory Management
<http://developer.apple.com/documentation/Cocoa/Conceptual/LanguageIntegration/Concepts/memory.html>

The Objective-C Programming Language (Xcode)
<file:///Developer/Documentation/Cocoa/Conceptual/ObjectiveC/index.html>

Cocoa-Dev (Apple) and MacOSX-Dev (Omni Group) mailing list archives
<http://cocoa.mamasam.com/>

ThinkfreeOffice

COMPATIBLE WITH MICROSOFT WORD, EXCEL AND POWERPOINT
WORDPROCESSOR • SPREADSHEET • PRESENTATION GRAPHICS

The Affordable Office Alternative!



Three High-Performance Applications

Thinkfree Write

Thinkfree Write is a powerful word processing application that enables you to create rich, professional quality documents and Web pages. You can insert tables, images, and clipart, or even apply custom layouts to your document...then effortlessly proofread your work with the easy-to-use spelling and auto-correction features.

Thinkfree Calc

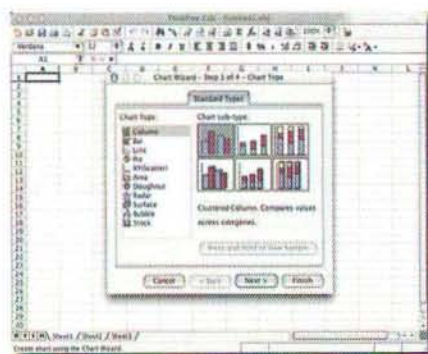
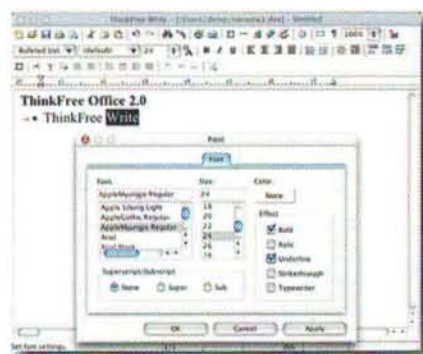
Thinkfree Calc is a full-featured, easy-to-use spreadsheet application that can easily tackle the most complex analytical tasks with over 40 charts and 300 function capabilities. Thinkfree Calc opens, edits, and saves directly into the Microsoft Excel (.xls) format, so users can seamlessly share documents and collaborate with Microsoft Office users.

Thinkfree Show

Thinkfree Show enables you to create high-impact presentations including animation effects, drawings, images, clipart, and other graphic features. Thinkfree Show opens, edits and saves directly into the Microsoft PowerPoint (.ppt) format.

CyberdrivePlus

A free, one-year subscription to CyberdrivePlus is also included. CyberdrivePlus provides you with secure, Internet file storage and free online software upgrades!



"Thinkfree is a best-of-breed program that will exceed your expectations."
— Jeffery Battersby



"Thinkfree Office is the next best thing and then some."
— Deborah Shadovitz



"Thinkfree Office is an impressive effort to crack the seemingly impenetrable productivity market."
— Chris Ward

amazon.com.

Apple Store



Apple Specialist



ONLY
\$49⁹⁵

By Rich Morin

ESR, etc.

A multiplex review of a singular individual...

Eric S. Raymond (aka ESR) is best known for his activities in support of Open Source software. Several years ago, he convinced a group of key developers to adopt the term as a less-ambiguous and more "marketable" replacement for the existing term, "free software".

Note: The term "Open Source" was actually coined by Christine Peterson (President of the Foresight Institute; <http://www.foresight.org>). Also, the term "Free Software" still has its adherents; see <http://www.fsf.org/philosophy>.

Since that time, Eric has written extensively on topics related to Open Source software. His essay "The Cathedral and the Bazaar" is probably the best known of these essays, but the others are also worth reading. See "The Cathedral and the Bazaar: Musings on Unix and Open Source by an Accidental Revolutionary" (O'Reilly; <http://www.oreilly.com>).

Two of Eric's essays, "A Brief History of Hackerdom" and "The Revenge of the Hackers" appear in the excellent collection "Open Sources: Voices from the Open Source Revolution" (DiBona, et al; O'Reilly). If you want to understand hacker culture and the Open Source movement, both of these O'Reilly books would be excellent starting points.

Eric has also been busy on the organizational and legal sides of Open Source. He is President of the Board of Directors of the Open Source Institute (<http://www.opensource.org>). He has also published (and publicized) a number of internal Microsoft documents, weighed in on relevant lawsuits (e.g., the SCO unpleasantness), etc.

OTHER PUBLICATIONS

Eric writes extensively, covering a wide range of topics. His home page (<http://catb.org/~esr>) has links to essays on anthropology, economics, politics, science fiction, and many other areas. Even if you don't agree with Eric's opinions, you should find his writings to be interesting and thought-provoking.

As the long-standing steward of the Jargon File (<http://www.jargon.org>; also available as "The New Hacker's

Dictionary" from MIT Press, <http://mitpress.mit.com>), Eric acts as an editor, lexicographer, historian, and (occasionally) anthropologist. The definitions tend to be far more interesting and enjoyable than one might expect from a conventional "dictionary".

THE ART OF UNIX PROGRAMMING

Over the past five years, Eric has been working on a distillation of the lessons that Unix has to offer the programming community. Taking his own advice ("Many eyes make all bugs shallow.") seriously, he brought in a number of Unix wizards as advisors. As a result, "The Art of Unix Programming" (Addison-Wesley; <http://www.aw.com>) is enjoyable, educational, and for the most part, authoritative.

Although the book doesn't address Mac OS X, in particular, it has quite a bit to say about the BSD side of the operating system. For instance, it talks at length about the Unix practice of encoding information in text files. OSX, with something like 100K text files, certainly seems to have adopted this idea.

It also discusses popular ideas such as object-oriented programming, threads, giving reasons why these might not be the best possible approach in all circumstances. All told, this book is the best introduction I can suggest for a Mac OS programmer who wants to understand the BSD (read, Unix) side of Mac OS X.

MORE CAN BE LESS

True to Unix tradition, Eric extols the benefits of having many different tools. Again, OSX is right there, providing a plethora of programming languages and other useful utilities. I wonder, however, whether OSX (and the Open Source community, in general) may not be suffering from an embarrassment of riches.

In perusing leads for programming positions, many of us have noted the extreme diversity of skills that may be demanded for a single position. Applicants are expected to have several years of experience in each of several languages and tools. Often, it is hard to conceive reasons for using these languages and tools in a single project.

Ignoring the discouraging effect of such postings on prospective applicants, let's consider how the company

Rich Morin has been using computers since 1970, Unix since 1983, and Mac-based Unix since 1986 (when he helped Apple create A/UX 1.0). When he isn't writing this column, Rich runs Prime Time Freeware (www.ptf.com), a publisher of books and CD-ROMs for the Free and Open Source software community. Feel free to write to Rich at rdm@ptf.com.

managed to get into the position of needing this sort of polymath. Speculating, I propose that the company once had a number of programmers, each developing part of a complex system. Each programmer chose the "right tool for the job", in true Unix fashion.

The company now has far fewer programming positions, but it still needs to maintain a polyglot mass of software. Their job postings are totally unrealistic – no single programmer can have extensive skills in more than a few areas – but they are merely a symptom of a larger problem.

OSX has similar historical baggage. If you look in `/etc`, you will find numerous control files. Most of these are "flat files", employing newlines to separate records and some form of delimiter (e.g., colons or "white space") to separate fields. Unfortunately, the exact format varies, so programs (and administrators) must "understand" different syntax for each file.

The log files are in even worse shape. They may be written by multiple programs, so the format can vary from entry to entry. Because the entries were written for human consumption, they often provide no unambiguous way for a program to distinguish fields. This is a serious problem; if log files are too big for humans to read, and too messy for programs to read, exactly what purpose are they serving?

Moving on to OSX's "plist" files, we find two distinct formats in evidence. One, inherited from NeXT, uses a C-like syntax. The other, strongly encouraged by Apple's current documentation, is based on XML. Using the appropriate frameworks, an Apple programmer can read either file. An administrator, however, may need to be familiar with both.

Although I'm a big fan of YAML (YAML Ain't Markup Language; <http://www.yaml.org>), I sometimes wonder if I might be making some programmer or administrator's life more difficult, by introducing yet another data format. Sigh.

Programming languages are another area of concern. OSX is built out of C, C++, Objective-C, and a smattering of scripting languages. C is used for the "Core OS" (kernel, BSD libraries and programs), but C++ is used for the IOKit and most device drivers. Objective-C is used for the higher-level frameworks and applications. Finally, the scripting languages are used for all sorts of administrative glue.

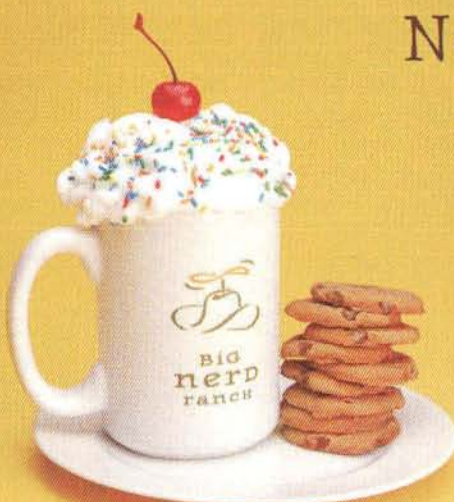
Fortunately, few programmers write programs in all these areas. If you're writing GUI-based apps, you're unlikely to be writing device drivers at the same time. If you're a serious scripter, you may never look at any C (etc) code. Unless, as I am, you're using Perl, Python, or Ruby to build Cocoa-based apps.

For all my misgivings, however, I wouldn't want to be restricted to a single data format or programming language. XML is powerful and very "buzzword-compliant", but it's also verbose and a poor match for associative arrays. Objective-C is a tidy integration of OO into the base C language, but its memory management seems a bit awkward to my Perl-accustomed eyes.

Ockham's Razor ("Pluralitas Non Est Ponenda Sine Necessitate.") transliterates to "Plurality should not be posited without necessity", reminding scientists to remain parsimonious in imagining new mechanisms. It is also translated, however, as "Entities are not to be multiplied beyond necessity" and even "Keep It Simple, Stupid".

In short, our challenge as developers is to decide how to achieve simplicity (once we have decided what "simplicity" is, in a given context :-), while retaining the power and flexibility that our mixed programming heritage provides.


Eric's latest offering, like many of his earlier writings, helps to show us the way...



Now serving Cocoa[®] just the way you want it.

Training for Mac OS X doesn't have to be the same old flavor.

- Reserve your seat in a class at our scenic lodge location, or have experts come to you for **Extreme Mentoring**. Two weeks of on-site instruction and collaboration, customized to the requirements of your project. Book now for 2003. See why we're different.



**Big
nerd
ranch**

Intensive Classes for Programmers
www.bignerdranch.com

by Tim Monroe

The Matrix Revolutions

Handling Movie File Operations with Revolution

INTRODUCTION

In the previous two *QuickTime Toolkit* articles (in *MacTech*, September and October 2003), we've taken a look at Revolution, a rapid application development tool published by Runtime Revolution Ltd. So far, we've developed a fairly complete application — called RunRevVeez — that can open and display QuickTime movies. Our application also supports basic editing operations on those movies (cut, copy, paste, and so forth). In this article, we'll finish up our investigation of Revolution by implementing the basic file-handling operations. We'll see how to keep track of whether a movie has been edited, how to adjust the items in the File menu accordingly, and how to save an edited movie into a new file. And we'll see how, when the user decides to quit RunRevVeez, to iterate through all open movie windows and give the user the chance to save or discard any unsaved changes to those movies. **Figure 1** shows the File menu of RunRevVeez when at least one movie window is open and the movie in the frontmost window has been edited.



Figure 1: The File menu of RunRevVeez

It's worth noting that these techniques might be of interest to any Revolution developers who need to create documents and store the information in them, as this issue does not appear to be well documented.

Toward the end of this article, we'll take a look at building and packaging RunRevVeez for distribution to users. The main hurdle here involves combining into a single item the application built by Revolution and the external plug-in module built by Project Builder. This isn't terribly difficult, but once again it's not well documented. So it's worth touching on the topic.

A few final notes before we begin. First, I have upgraded my development environment to Revolution version 2.1, which is the current version as of the time I'm writing this. (The two previous articles used version 2.0.2.) So you may notice some differences in the interface if you're still using an earlier version. I encountered no problems when upgrading to version 2.1; the existing version 2.0.2 RunRevVeez project could be opened and modified using version 2.1. Moreover, by the time this article makes it to press, the current version will be at least 2.1.1 (more on that later). The folks at Runtime Revolution seem to be committed to making their Mac OS X product as solid and stable as they can, and improved versions seem to be appearing fairly often. This is a good thing.

Also, you may notice that the movie windows and dialog boxes have a slightly different appearance from those in the two previous articles. That's because the screen shots for this article were taken with RunRevVeez running on Mac OS X version 10.3 (also known as "Panther"). I encountered no problems running Revolution or the applications it creates under Panther. This also is a good thing.

FILE MANIPULATION

So our goal right now is to finish RunRevVeez by implementing the basic file-handling operations (aside from the Open menu item, which we covered in the earlier articles). In particular, we want to be able to track changes to a movie and allow the user to save those changes. In the previous article, you'll recall, we saw how to implement the standard editing operations. To do this, we needed to use an external code module that called QuickTime's movie controller editing APIs (for instance, `MCCut` and `MCPaste`). We also defined a custom property for each movie window (which we called

Tim Monroe is a member of the QuickTime engineering team at Apple. You can contact him at monroe@mactech.com. The views expressed here are not necessarily shared by his employer.

movieChanged) to keep track of whether the movie in that window has been edited.

Enabling and Disabling the File Menu Items

We can use that custom property to help us enable and disable the items in the File menu according to the state of the movie in the frontmost window. We want the New and Open menu items to be enabled always, and we want the Close and "Save As..." menu items to be enabled only if the frontmost window is a movie window. Finally, we want the Save menu item to be enabled only if the frontmost window is a movie window that has been changed since it was last opened or saved. **Listing 1** shows the code we add to the mouseDown handler of the main menu item group.

Listing 1: Adjusting the File menu

```
on mouseDown
    constant kNewItemIndex = 1
    constant kOpenItemIndex = 2
    constant kCloseItemIndex = 3
    constant kSaveItemIndex = 5
    constant kSaveAsItemIndex = 6

    put first line of the openStacks into theTopStack
    put exists(player "MoviePlayer" of stack theTopStack) \
        into gotPlayer

    enable menuItem kNewItemIndex of menu "File"
    enable menuItem kOpenItemIndex of menu "File"

    disable menuItem kCloseItemIndex of menu "File"
    disable menuItem kSaveItemIndex of menu "File"
    disable menuItem kSaveAsItemIndex of menu "File"

    if gotPlayer then
        enable menuItem kCloseItemIndex of menu "File"
        enable menuItem kSaveAsItemIndex of menu "File"

        if the movieChanged of stack theTopStack is true then
            enable menuItem kSaveItemIndex of menu "File"
        end if
    end if
end mouseDown
```

There's nothing here that we haven't seen before, except the use of the **constant** command to define some symbolic constants. This of course makes our code more readable and (in theory) more maintainable.

Creating a New Movie

In the previous two articles, we saw how to handle the Open command in the File menu to open an existing movie file and to display the movie it contains in a window on the screen. Revolution provides easy access to the standard file-opening dialog box, and it returns the full pathname of a selected movie file, which we can assign to the movie player object.

But how would we handle the New menu item, which should open a new movie window that contains an empty movie not associated with any existing file? Here things get a bit tricky. We can open an empty document window and leave the filename of the player object in that window set to the empty string "". But in that case the movieControllerID property of the



Fetch

For Mac OS X, 9, 8, 7...
(Running dog included.)

X

Fetchsoftworks.com

Version 4.0.3 now available.

player object will be 0, and this renders the new player object pretty much useless. We won't be able to paste any movie data cut or copied from some other movie into it (which is usually what we want to do with empty movies).

One solution to this problem would be to create a new empty movie and an associated movie controller ourselves (in our QuickTime external module, of course) and then assign the movie controller ID to the player object's `movieControllerID` property. The Revolution documentation states unequivocally that "the `movieControllerID` property is read-only and cannot be set". However, I'm told that this is in fact not true and that we *can* assign a value to that property. Because I learned about this fairly late in the process of writing this article, I'll have to leave the implementation of this feature as an exercise for the reader. In the meantime, RunRevVeez will simply display a dialog box, shown in **Figure 2**, when the user selects the New menu item.

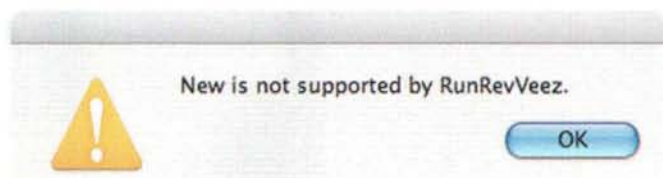


Figure 2: The New warning

Closing a Movie Window

When the user clicks the close button of a stack, Revolution does not immediately close the stack. Instead, it sends a `closeStackRequest` message to the current card on that stack. If the `closeStackRequest` message handler passes that message further along the message path, the stack will actually be closed; otherwise, the stack remains open. This mechanism gives us a chance to prompt the user to save or discard changes to the movie in a window, or to cancel the close operation altogether.

We also want to make use of this mechanism when the user selects the Close menu item in the File menu. Accordingly, we'll have the `menuPick` handler for that item simply send a `closeStackRequest` message to the frontmost movie window.

Listing 2 shows our code that handles the Close menu item.

Listing 2: Handling the Close menu item

```

case "Close"
    put first line of the openStacks into theTopStack
    if exists(player "MoviePlayer" of stack theTopStack) then
        send closeStackRequest to stack theTopStack
    end if
    break
menuPick

```

So we need to add a `closeStackRequest` handler to the script associated with a movie window. This handler needs to check the `movieChanged` property of the movie window to see if the movie has been edited since it was last opened or saved. If it has been, we want to ask the user to save those changes, discard

them, or cancel the close operation. We can do that with this lengthy line of script:

```

answer warning "Do you want to save the changes \
you made to the document " & quote & the title of me \
& quote & "?" \
with "Don't Save" or "Cancel" or "Save" \
titled "Save changes before " & theAction \
as sheet

```

This line displays the sheet shown in **Figure 3**.

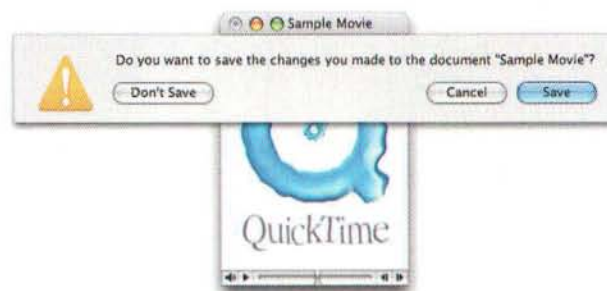


Figure 3: The "Save changes" sheet

A couple of comments are in order here. First, notice that we use the special keyword `quote` to embed a pair of quotation marks in the string displayed as the message in the sheet. Also, we use the "as sheet" qualifier to have the dialog box drop down from the movie window as a sheet. On operating systems other than Mac OS X, this qualifier is ignored and the warning is displayed as a standard modal dialog box. In that case, the dialog box will have the specified title, which is either "Save changes before closing" or "Save changes before quitting". We look at the custom property `isQuitting` of the mainstack to determine which action is appropriate:

```

if the isQuitting of stack "RunRevVeez" is true then
    put "quitting" into theAction
else
    put "closing" into theAction
end if

```

(We'll see how that property is set in a little while.)

The user must click one of the three buttons in the sheet (or dialog box) to dismiss it; when that happens, control returns to our `closeStackRequest` handler and the `it` variable is set to the text of the selected button. If the user selects "Don't Save", we can simply pass the `closeStackRequest` message along the message path, thereby allowing the window to be closed. If the user selects "Save", we call the `saveAs` function, which is defined in our external code module. (See the following two sections for more on saving edited movies.) Finally, if the user selects "Cancel", we want to set the mainstack's `isQuitting` property to `false` so that RunRevVeez does not quit. (If we weren't already quitting, this is harmless.) Also, we want to execute the "exit to top" control

32
< 1 year
2 years >
\$64



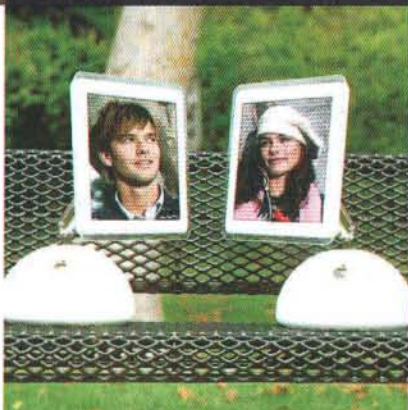
INTERVIEWS

TAPPING INTO THE WORLD OF CELEBRITIES AND THEIR MACS, ONLY MACDIRECTORY OFFERS EXCLUSIVE INTERVIEWS. GET A CLOSE AND PERSONAL VIEW FROM SARAH JESSICA PARKER, STING, STEVE JOBS, MADONNA, HARRY CONNICK JR., GEORGE LUCAS, JENNIFER JASON LEIGH, STEVE WOZ AND OTHER LEADERS IN THE MAC COMMUNITY.



FEATURES

DESIGNERS, WRITERS, MUSICIANS, BUSINESS LEADERS & OUR TECHNICAL EXPERT TEAM OFFER THEIR OWN PERSONAL INTERPRETATION OF THINGS THAT ONLY THE MAC SYSTEM CAN DELIVER. FEATURING OVER 240 PAGES OF REVIEWS, INTERVIEWS, NEWS, INSIGHTS, TRENDS AND THE LARGEST MACINTOSH BUYERS' GUIDE INCLUDING OVER 5,000 MAC PRODUCTS AND SERVICES.



CULTURE

MACDIRECTORY TAKES YOU TO THE WILDEST CORNERS OF THE WORLD AND UNCOVERS HOW MACINTOSH COMPUTERS ARE BEING USED BY OTHER CULTURES. TRAVEL TO JAPAN, AUSTRALIA, GERMANY, BRAZIL, INDIA, RUSSIA & LEARN MORE ABOUT APPLE'S CULTURAL IMPACT AROUND THE GLOBE.

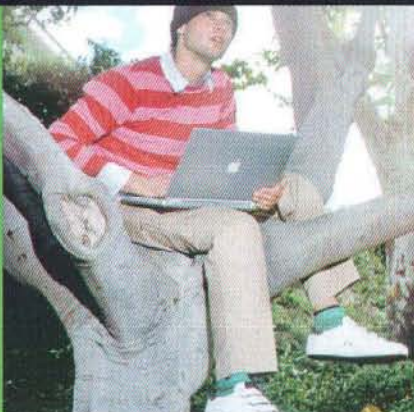
MacDirectory

BEYOND ANY MACINTOSH MAGAZINE. SUBSCRIBE.

< Subscribe

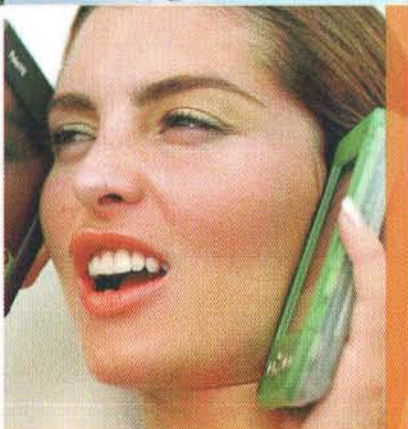
www.macdirectory.com/mw.html

SEND CHECK OR MONEY ORDER TO:
MACDIRECTORY SUB DEPT.
326 A STREET, 2C
SOUTH BOSTON, MA 02110



REVIEWS

FIND OUT ALL YOU NEED TO KNOW ABOUT THE LATEST MAC PRODUCTS INCLUDING THE HOTTEST MAC OS SOFTWARE AND HARDWARE.



WIN!

SUBSCRIBE TO MACDIRECTORY AND YOU WILL AUTOMATICALLY ENTER OUR SWEEPSTAKES FOR A CHANCE TO WIN A NEW TITANIUM!

statement to jump out of the `closeStackRequest` message handler and all other pending handlers. As we'll see in more detail soon, this will halt a `shutdownRequest` handler that might be executing.

Listing 3 shows our complete `closeStackRequest` message handler.

Listing 3: Handling a request to close a window

```

closeStackRequest
on closeStackRequest
  if the movieChanged of this stack is true then

    - figure out whether we are quitting or closing
    if the isQuitting of stack "RunRevVeez" is true then
      put "quitting" into theAction
    else
      put "closing" into theAction
    end if

    - ask the user to save or discard changes, or to cancel the close operation
    answer warning "Do you want to save the changes \
you made to the document " & quote & the title of me \
& quote & "?" \
with "Don't Save" or "Cancel" or "Save" \
titled "Save changes before " & theAction \
as sheet

    - the button chosen by the user is now in the "it" variable; handle the three cases
    - Cancel:
    if it is "Cancel" then
      set the isQuitting of stack "RunRevVeez" to false
      exit to top
    end if

    - Save: save the changes and close window
    if it is "Save" then
      put the movieControllerID of player "MoviePlayer" \
of this stack into mc
      get saveAs(mc)
    end if

    - Don't Save: toss the changes and close window
    - (no actual code needed here)

  end if

  - if we get to here, we want to close the window
  close me
end closeStackRequest

```

In theory (at least as I understand it), we should use the command "pass `closeStackRequest`" instead of "close me" near the end of this handler, to pass the message along the message chain. But I couldn't get things working correctly if I did that. At any rate, this handler appears to do the right thing.

Saving a Changed Movie

Once the user has edited a movie, he or she is likely to want to save those changes. Unfortunately, the current version of Revolution (version 2.1) makes it impossible for us to save an edited movie into the file the movie was opened from. That's because, when the Revolution runtime engine calls `OpenMovieFile` to open a movie file, it passes the value `fsRdPerm` as the desired file permission. That is, Revolution always opens movie files with *read-only* permission. So, even though `RunRevVeez` is able to actually edit a movie, it can't save the edited movie back into its original file.

The engineers at Runtime Revolution are aware of this limitation and have indicated to me that version 2.1.1 will be able to open movie files with *read/write* permission. In addition, Revolution would need to provide accessor methods that return a movie's file reference number (which QuickTime returns to `OpenMovieFile`) and its resource ID (which QuickTime returns to `NewMovieFromFile`), since we would need both those values when we call `UpdateMovieResource`. Because version 2.1.1 is not yet available, I cannot verify that it provides the capabilities we need to save an edited movie into its original file. As a result, `RunRevVeez` simply displays a warning when the user selects the Save menu item. It executes this line of script to do so:

```

answer warning "Save is not supported by RunRevVeez. Use \
Save As...."

```

Figure 4 shows the resulting warning.

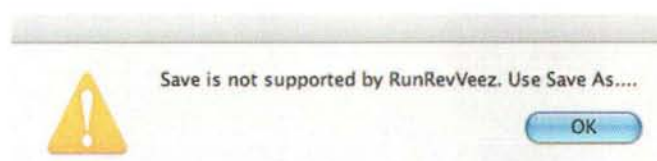


Figure 4: The Save warning

Saving a Movie into a New File

The best we can do right now, therefore, is allow the user to save an edited movie into a new file — that is, implement the "Save As..." menu item. Our menu handler for the "Save As..." item begins as usual by finding the frontmost window (that is, stack) and ensuring that it contains a player object. Then it calls the custom function `saveAs`:

```

put the movieControllerID of player "MoviePlayer" \
of stack theTopStack into mc
get saveAs(mc)

```

The code for the `saveAs` function is contained in our QuickTime external; **Listing 4** shows the corresponding external code, `XCMD_SaveAs`. The central core of this function is borrowed wholesale from our existing C-language sample code application `QTShell`, as are the utility functions `QTFrame_PutFile` and `QTUtils_ConvertCToPascalString` that are called by `XCMD_SaveAs`.

Listing 4: Saving a movie in a new file

```

XCMD_SaveAs
#define kSavePrompt          "Save Movie as:"
#define kSaveFileName        "untitled.mov"

void XCMD_SaveAs (char *args[], int nargs,
                  char **retstring, Bool *pass, Bool *error)
{
  MovieController mc = NULL;
  Movie movie = NULL;
  char *nufilename = NULL;
  OSErr result = userCanceledErr;
  char *retstr = NULL;

```



```

*pass = false;
*error = false;
if (nargs == 1) {
    mc = (MovieController)atol(args[0]);
    if (mc != NULL) {
        movie = MCGetMovie(mc);
        if (movie != NULL) {
            FSSpec mfile;
            Boolean isSelected = false;
            Boolean isReplacing = false;
            StringPtr prompt =
                QTUtils_ConvertCToPascalString(kSavePrompt);
            StringPtr fileName =
                QTUtils_ConvertCToPascalString(kSaveFileName);

            QTFrame_PutFile(prompt, fileName, &mfile,
                           &isSelected, &isReplacing);

            free(prompt);
            free(fileName);

            // save the movie in the selected location
            if (isSelected) {
                Movie    newMovie = NULL;
                short    refNum = -1;
                FSRef     fsRef;

                // delete any existing file of that name
                if (isReplacing) {
                    result = DeleteMovieFile(&mfile);
                    if (result != noErr)
                        goto bail;
                }

                newMovie = FlattenMovieData(movie,
                    flattenAddMovieToDataFork |
                    flattenForceMovieResourceBeforeMovieData,
                    &mfile,
                    FOUR_CHAR_CODE('TVOD'),
                    smSystemScript,
                    createMovieFileDeleteCurFile |
                    createMovieFileDontCreateResFile);
                result = GetMoviesError();
                if ((newMovie == NULL) || (result != noErr))
                    goto bail;

                // FlattenMovieData creates a new movie file and returns the movie to us;
                // since we want to let Revolution open the new file, we'll dump the
                // movie returned by FlattenMovieData
                DisposeMovie(newMovie);

                // also, on MacOS, FlattenMovieData "always" creates a resource fork;
                // delete the resource fork now....

                #if TARGET_OS_MAC
                result = FSpOpenRF(&mfile, fsRdWrPerm, &refNum);
                if (result == noErr) {
                    SetEOF(refNum, 0L);
                    FSClose(refNum);
                }
            }

            // get the full pathname of the new file (to pass back to caller)
            result = FSpMakeFSRef(&mfile, &fsRef);
            if (result == noErr) {
                retstr = malloc(kMaxPathSize);
                result = FSRefMakePath(&fsRef, retstr,
                                       kMaxPathSize);

                if (result == noErr) {
                    *retstring = retstr;
                    return;
                }
            }
        }
    }

    bail:
    retstr = calloc(1, 2);
    if (retstr != NULL)
        retstr[0] = (result == noErr) ? '0': '1';
}

```

```

*retstring = retstr;
}

```

The typical behavior of the "Save As..." menu item is to create (or overwrite) the target movie file and then to open the movie in that file in the existing movie window. That means that we need to get RunRevVeez to open that new (or overwritten) file and load its movie into the movie window. To do this, we need to know the full pathname of that file. If you look carefully, you'll see that `XCMD_SaveAs` calls `FSpMakeFSRef` and `FSRefMakePath` to convert the file system specification of the target file into a full pathname, and that this pathname is returned (in `retstring`) to the caller. In other words, if the call to `saveAs` completes successfully, the built-in variable it will contain the full pathname of the new (or overwritten) file; if that call fails for any reason, then it will be the empty string or set to "1".

If the call to `saveAs` succeeds, we need to open the movie in the target file, as shown in **Listing 5**.

Listing 5: Loading the target movie into a movie window menuPick

```

- get the base name of the file
set the itemDelimiter to "/"
put the last item of it into newStackName

set the filename of player "MoviePlayer" \
    of stack theTopStack to it
set the title of stack theTopStack to newStackName

```

stockicons.com

brought to you by the Iconfactory

Icons to go!

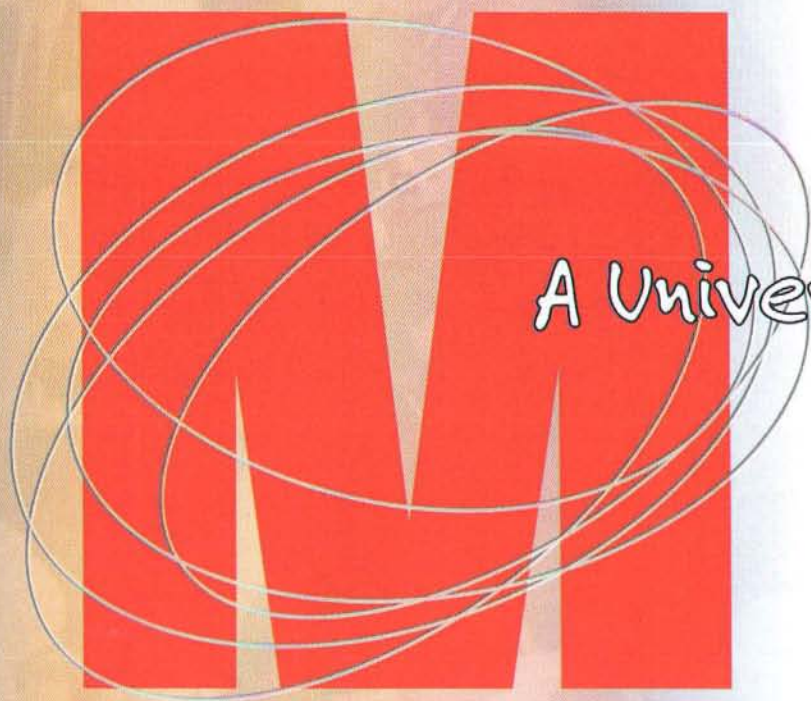


Now developers can purchase complete collections of professionally designed icons at an affordable price from the premier source for custom designed icons in the industry. See some of the work we have done for Apple, Microsoft, Aladdin, Intuit, Palm and many others at www.iconfactorydesign.com.

\$349.00

Each stock icon collection contains 80 individual icons in three pixel sizes - 32x32, 24x24 and 16x16. Collections are unique in style and are provided in an array of formats including transparent TIFFs, GIF, PNG, .icns, and Windows format .ico's.

For more information visit www.stockicons.com



A Universe of Solutions

Enhance your knowledge and skills by attending the world's most comprehensive forum for Mac users. Transcend your boundaries and make informed purchasing decisions.

January 5-9, 2004

San Francisco, CA
The Moscone Center

Conference January 5-9, 2004
Expo January 6-9, 2004



Macworld
Conference & Expo®

Find out what all the buzz is about at
www.macworldexpo.com

IDG
WORLD EXPO

Flagship Sponsors:

Macworld

Macworld Conference & Expo,[®] 20 years young, will deliver you a universe of solutions for all your Macintosh needs.

Don't miss your opportunity to experience everything Mac in one place!

- ◆ Acquire knowledge to stay competitive
- ◆ Test drive new and innovative products
- ◆ Network with like-minded people
- ◆ Talk with the companies that make you more productive
- ◆ Feel the power of this holistic community
- ◆ Enjoy the atmosphere

Register online with
Priority Code: **A-MTD**

Macworld.com  **MacCentral**




```
set the movieChanged of stack theTopStack to false
get windowSetModified(windowID of stack theTopStack, 0)
```

We've seen much of this code before, since we needed to perform these operations when we opened a movie file into a new window. Notice that we reset the `movieChanged` property to false and call `windowSetModified` to clear the window modification state.

Quitting the Application

As you know, on Mac OS X the Quit menu item is contained in the Application menu, as shown in **Figure 5**.



Figure 5: The Application menu

This menu is supplied by the operating system, so we cannot attach a `menuPick` script to it. Instead, when the user selects the Quit menu item, the operating system sends an Apple event to our application. Since we have not installed any Apple event handlers, the Revolution runtime engine handles that event and issues a `shutDownRequest` message to our application. So we can handle the Quit menu item either by installing an Apple event handler or a `shutDownRequest` handler. To facilitate moving RunRevVeez to Windows operating systems, let's use a `shutDownRequest` handler.

We'll begin our `shutDownRequest` handler by setting a custom property of the mainstack, `isQuitting`:

```
set the isQuitting of me to true
```

We've already seen where we need to inspect this property, in the `closeStackRequest` handler of the movie window (**Listing 3**).

The remainder of our `shutDownRequest` handler simply loops through all open movie windows (using the `repeat` control structure) and sends the `closeStackRequest` message to them. If we encounter a window that isn't a movie window, we simply close it. Revolution applications will quit automatically once the last open window is closed. **Listing 6** shows our complete `shutDownRequest` handler.

Listing 6: Handling the Quit menu item

```
on shutDownRequest
    set the isQuitting of me to true
    shutDownRequest
```

```
- check for "dirty" movie windows
repeat for each line theStack in the openStacks
    if exists(player "MoviePlayer" of stack theStack) then
        - ask the movie window to close
        send closeStackRequest to stack theStack
    else
        - if it's not a movie window, just close it
        close stack theStack
    end if
end repeat

pass shutDownRequest

end shutDownRequest
```

APPLICATION DISTRIBUTION

Let's finish up by considering how to link our external code module — which contains virtually all of our QuickTime-specific code — to our application RunRevVeez. During application development, we can do this by setting the External References property of the mainstack to the bundle built by Project Builder. Recall that Project Builder builds a module called `QTExternal.bundle`. I found it most useful to copy that bundle into the folder containing the Revolution IDE. Then I added the full pathname of that bundle to the list of external references, as shown in **Figure 6**. The easiest way to set this external reference is to click on the little folder icon and then navigate to the desired file.

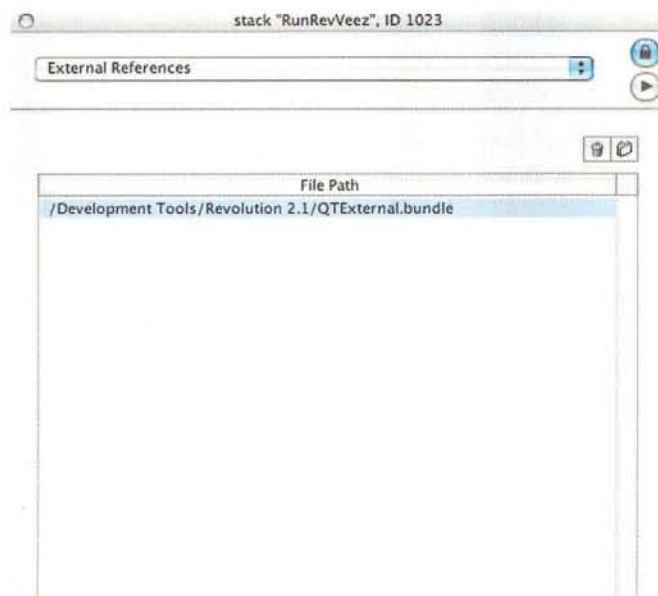


Figure 6: The external references during application development

Thereafter, whenever RunRevVeez is executed (either as a standalone application or in the Revolution IDE itself), the

specified bundle will be used to resolve the external references in RunRevVeez.

Obviously, however, this is not satisfactory as a final distribution method. It would be far better to package the external and the application into a single file that can be installed wherever the user wishes, copied from machine to machine, and so forth. Once again, we want to set the External References property of the mainstack, but this time to a path relative to the compiled application. The easiest way to do that is using the *message window* (or *message box*). The message window is a window that allows us to execute one or more lines of script without having to create a message handler or attach the handler to any object. **Figure 7** shows the Revolution message window, with the appropriate command typed into it.

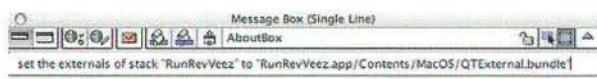


Figure 7: The message window

If we inspect the External References property of the mainstack once again, we'll see the desired relative path (**Figure 8**).

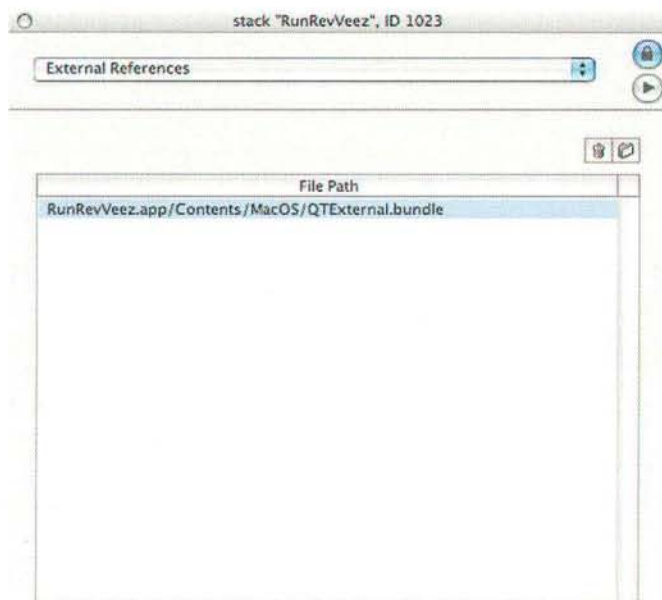


Figure 8: The external references for application distribution

All that remains is to copy the external bundle into the application bundle. Once we've got RunRevVeez working as desired and we've reset the External References property as just indicated, option-click the compiled application in the



THE LAW OFFICE OF BRADLEY M. SNIDERMAN

Need help safeguarding your software?

If you're developing software, you need your valuable work protected with trademark and copyright registration, as well as Non Disclosure Agreements.

Then, when you are ready to sell it, you can protect yourself further with a licensing agreement.

I am an attorney practicing in Intellectual Property, Business Formations, Corporate, Commercial and Contract law.

Please give me a call or an e-mail. Reasonable fees.

23679 Calabasas Rd. #558 • Calabasas, CA 91302
PHONE 818-222-0365 FAX 818-591-1038 EMAIL brad@sniderman.com

Finder and choose "Show Package Contents", as shown in **Figure 9**.

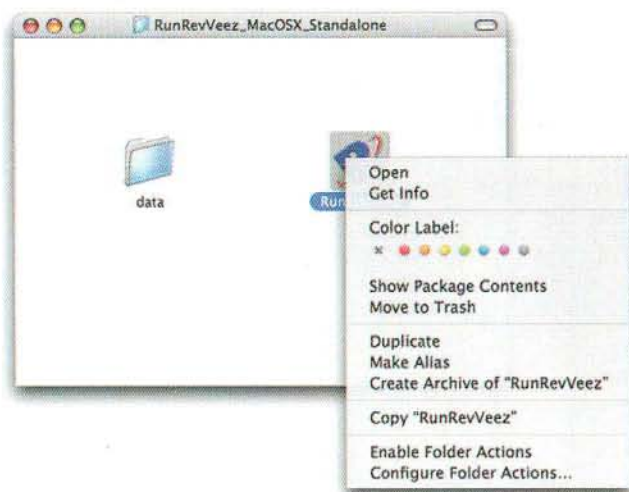


Figure 9: The contextual menu for RunRevVeez

Finally, copy the `QTExternal.bundle` file into the MacOS folder that's inside the Contents folder of the window that opens up. We're done.

CONCLUSION

Let's take stock of what we've learned about Revolution and QuickTime in these three articles. We set ourselves a very specific goal: duplicate the functionality of our existing Carbon application `QTShell` using Revolution. That meant: create an application capable of opening arbitrary QuickTime movie files in windows on the screen, support the standard movie editing operations, and allow the user to save any edited movies as desired. That also meant: have our application look and behave like a standard double-clickable application (using native controls and other widgets provided by the operating system), support an About box, and implement the typical menus and menu items for movie windows and documents.

Revolution was able to provide most of what we were looking for, especially since we can take advantage of its ability to call external code modules. For instance, Revolution does not provide built-in support for movie editing, but it was easy enough for us to add that ability by defining a few routines in an external module. As I said in an earlier article, writing and deploying an external module is so easy that I can't really imagine any serious developer using Revolution not wanting to employ them as a matter of course. In an external, we have access to virtually the entire set of QuickTime APIs, which we can bring into play as needed.

Well, almost. There are a few limitations to using QuickTime APIs within Revolution applications that are worth highlighting. We saw earlier in this article that the Revolution runtime engine

opens movie files with read-only permission. This means that we cannot save an edited movie into the file it was originally opened from. As we saw, however, this does not mean that we cannot usefully edit movies using RunRevVeez; rather, we simply have to educate the user to save the edited movie into a new file. The staff at Runtime Revolution has indicated that a fix for this issue is in the works, so happily (by the time you read this) our "Save As..." workaround may no longer be necessary.

A more significant limitation is that the Revolution runtime engine installs a movie controller action filter procedure. I'm guessing that it does this to support the messages that can be sent to a player object when various events concerning the QuickTime movie occur. For example, when the user changes the movie's current time (perhaps by clicking in the time line area of the controller bar, or by dragging the thumb), a `currentTimeChanged` message is sent to the player object. Similarly, for a QuickTime VR movie, the Revolution runtime engine installs an intercept procedure so that it can issue `hotSpotClicked` messages at the appropriate time.

The trouble (as we learned in an earlier article on REALbasic) is that this effectively prevents our external code module from installing a movie controller action filter procedure (or QuickTime VR intercept procedure) of its own. This then limits our ability to take advantage of a large number of capabilities that rely on an application processing messages in a movie controller action filter procedure. (We've seen very many examples of this in earlier articles.)

Let's be very clear about this, however: Revolution isn't doing anything wrong by installing and using these callback procedures. Rather, it's providing functionality to its users in exactly the way it should. The problem arises because Apple has as yet provided no means of chaining movie controller action filter procedures or QuickTime VR intercept procedures. One way out of this quandary would be for Apple to do exactly that: provide a way for multiple filter procedures to be called by a single movie controller. It might also be possible for Revolution to support an "expert mode" that allowed an application to selectively turn off the installation of these procedures. That would allow an external module to take full advantage of these various callback procedures without stepping on Revolution's toes.

So let's wrap thing up: RunRevVeez now looks and behaves exactly as it should, with only minor exceptions. It can open and display QuickTime movies, allow the standard editing operations on those movies, and save the edited movies into new files. It gives the user an opportunity to save or discard edited movies when a window is about to close or when the application is about to quit. And all of this was achieved with a couple dozen lines of script and an easy-to-construct external code module.

CREDITS

Thanks are due once again to Tuviah Snyder at Runtime Revolution Ltd., who was especially helpful this time in figuring out several issues concerning the packaging of our QuickTime external module.

Ready to Build a Better MAC?

**Easy to Install
upgrades help you...**

- 1. Add more *HD* space**
- 2. Add a faster *CPU***
- 3. Add more *RAM***
- 4. Add new *ports***

**We know Macintosh
Technology.**

**Come to DevDepot for all
the cool upgrades!**

**DEV
DEPOT®**

877-DEPOT-NOW

www.devdepot.com/bettermac



By Kevin Hemenway, *Titillating Entertainer*

Panther Roars, Database Meows

Panther worth your time? Databases besides Filemaker? Pffffttt.

"What's new in Panther?" you shout with mindless adulation, anxious saliva and drool waiting to soil the pages of your newest MacTech. Bad news, bub: absolutely nothing: nothing to write about, nothing to justify your upgrading, nothing worth your time, effort, or worrisome heart palpitations. I was hoping I could pound out a peck of pickled pages on pathetic Panther proselytizing, but no, Apple has spurned my deadly Shaolin technique.

I am, of course, talking specifically about web serving.

MR. PISER, I THINK YOU SHOULD COME UP HERE

Sure, a lot of *other* incredible drool-worthy stuff has happened in the latest OS: Exposé, word completion in certain applications (via Option-Escape or F5), FileVault for the paranoids, gcc 3.3 and distributed building, a finished version of X11, Xcode, blah, blah, blah. But web serving? Nothing, nadda, zilch. Happily, this is not really a negative—it's hard to improve on something that was nearly perfect in the first place (of course, this all depends on your needs, desires, and expectations).

Which isn't to say there's nothing new with the Apache web server and its related technology... just nothing monumental, merely incremental, ornamental, and supplemental. I'd have loved to see Apache 2.0 and MySQL 4.0, but let's cover the relevant changes we got instead. Note that I'll only cover areas we've touched upon in the previous columns: explaining the differences between features we've yet to use isn't very productive. When the time comes, if there are crucial differences between Jaguar and Panther, I'll address them.

- Apache is now version 1.3.28; the latest available for Jaguar was 1.3.27. 1.3.28 is mostly a security and bugfix release so, although welcome, there's nothing to really get excited

about. If you're interested in the specifics and technicalities, you can read the sordid details at http://www.apache.org/dist/httpd/CHANGES_1.3 (which also lists modifications for the just-released, as of this writing, 1.3.29).

- The output of `httpd -V` (which shows what Apache has been compiled with) includes a new entry: `-D DYNAMIC_MODULE_LIMIT=64`. This is a standard Apache compile-time setting, and controls how many modules can be dynamically loaded (fairly obvious, eh?). "64" is the standard Apache default; its explicit definition here is unnecessary (but there's no harm in doing so). More info at http://httpd.apache.org/dev/apidoc/apidoc_DYNAMIC_MODULE_LIMIT.html.
- Our included PHP is now version 4.3.2 (as `grep PHP /var/log/httpd/error_log` confirms). Even though 4.3.4 or higher will probably be available by the time you read this, a number of good security, performance, and language enhancements were made from 4.1.2 (the latest Jaguar version) onward. Most notably, PEAR (the "PHP Extension and Application Repository") is now part of PHP by default, although it seems to be broken in 10.3—see "Homework Malignments" below for analysis.

The PHP configuration has been improved in `/etc/httpd.conf`. Under Apache 1.3.27, we had the `AddModule` and `LoadModule` lines, as well as `AddType`'s for `.php` and `.phps` (see last month's *Untangling the Web*). We've got the same thing in 1.3.28, but they've been slightly improved (**Listing 1**). First, the MIME types are more tightly defined—they'll only come into play when the PHP module is loaded (`IfModule` is a simple conditional... *if* `mod_php` was loaded, *then* do this). The second improvement is the addition of `index.php` to the `DirectoryIndex`, which allows that file to be served by default when someone requests a directory (<http://disobey.com/example/>).

Kevin Hemenway, coauthor of *Mac OS X Hacks* and *Spidering Hacks*, is better known as Morbus Iff, the creator of disobey.com, which bills itself as "content for the discontented." Publisher and developer of more home cooking than you could ever imagine (like the popular open-sourced aggregator AmphetDesk, the best-kept gaming secret Gamegrene.com, the ever ignorable Nonsense Network, etc.), he is absolutely and positively disgusted by the lack of Jolt cola within walking distance. Contact him at morbus@disobey.com.



**ENGINEERING
BUSINESS
SOLUTIONS**

- SINCE 1989 -

PREMIER REPORT SOLUTIONS FOR MAC OS X

VVI®

www.vvi.com

info@vvi.com

888-VVI-PLOT

- As you'd expect from a new PHP install, the `<? phpinfo(); ?>` we learned about last issue is different (**Figure 1**). Along with a heavier command line that explicitly enables a few features, the configuration file (`php.ini`) has moved from `/usr/lib/` to `/etc/`, an arguably smarter location. If you've installed the recommended `php.ini` (<http://cvs.php.net/co.php/php-src/php.ini-recommended>), be sure to move it and restart Apache (`sudo apachectl restart`). Even with the upgrade, the default Apple-supplied PHP doesn't come close to the number of features and built-ins provided by Liyanage's version (<http://www.entropy.ch/software/macosx/php/>) or Server Logistics' Complete PHP (<http://www.serverlogistics.com/php4.php>).
- Perl, which we talked about during our two CGI articles, has been upgraded to 5.8.1-RC3; the final version of 5.8.1 has since been released. If you already know a thing or two about Perl, be forewarned that any modules you've compiled under Jaguar's 5.6.0 (like `XML::Parser`, `HTML::Parser`, `Compress::Zlib`, etc.) will need to be reinstalled, as the two different Perls are not binary compatible. Also note that `cpan`, similar to `perl -MCPAN -esomething`, is also available (find out more with `man cpan`).

Listing 1: Configuring PHP under Panther

The improved PHP configuration from `httpd.conf`

```
<IfModule mod_php4.c>
# If php is turned on, we respect .php and .phps files.
AddType application/x-httpd-php .php
AddType application/x-httpd-php-source .phps

# Since most users will want index.php to work we
# also automatically enable index.php
<IfModule mod_dir.c>
    DirectoryIndex index.html index.php
</IfModule>
</IfModule>
```

System	Darwin MostTowallets.local 7.0.0 Darwin Kernel Version 7.0.0: Wed Sep 24 15:48:39 PDT 2003; root:xnu-517.obj-1/RELEASE_PPC Power Macintosh
Build Date	Sep 13 2003 22:00:43
Configure Command	'./configure' '--prefix=/usr' '--mandir=/usr/share/man' '--infodir=/usr/share/info' '--with-aps' '--with-ldap=/usr' '--with-kerberos=/usr' '--enable-ctype' '--with-zlib-dir=/usr' '--enable-trans-sid' '--with-xm' '--enable-xml' '--enable-ftp' '--enable-mbstring' '--enable-dbx' '--enable-sockets' '--with-iodbc=/usr' '--with-curl=/usr' '--with-config-file-path=/etc'
Server API	Apache
Virtual Directory Support	disabled
Configuration File (php.ini) Path	/etc
PHP API	20020918
PHP Extension	20020429
Zend Extension	20021010
Debug Build	no
Thread Safety	disabled

Figure 1: The output of PHP 4.3.2's `phpinfo()` under Panther.

WAIT, WAIT, HOLD UP DOG... MySQL?

"Apache 2.0 and MySQL 4.0?" What's this MySQL (<http://www.mysql.com/>) thingy thing? Quite simply, it's your new Filemaker or 4D, only without the pretty pictures. Pronounced *my-ess-que-ell*, it's one of the most popular free and open sourced databases, installed and readily available to most web hosting accounts you'd normally pay monthly fees for. It's also installed by default in Apple's Mac OS X Server. Another, arguably stronger, database program is PostgreSQL (<http://www.pgsql.com/>) but, in a move sure to flirt with disaster, I'll rudely ignore it for most of my *Untangling the Web* columns. Enough emails convincing me otherwise will certainly sway my decision.

Since this column assumes you're **not** using OS X Server, our first step is to install MySQL. Incidentally, this is also the most complicated, not because it's difficult, but because there are so many different directions in which to approach it:

- **Compile the Source Code:** The geekiest among us will take MySQL's raw source code and compile it from scratch. This is the approach taken by Jay Greenspan in his Apple Internet Developer article (<http://developer.apple.com/internet/macosx/osdb.html>). Incidentally, I offered the same approach in a similar web-serving series for 10.1: http://www.macdevcenter.com/pub/a/mac/2002/03/08/apache_mac_5.html. While compiling from source gives you ultimate control, I've yet to see a situation that wasn't adequately handled by using a pre-packaged version (below).
- **Install the Official Binary:** MySQL provides Mac OS X double-clickable installers in "Standard", "Max", or "Debug" varieties (<http://www.mysql.com/downloads/mysql-4.0.html>). "Standard" is recommended for most, "Max" contains features that have yet to be fully tested, and "Debug" is suicidal for production environments. If you're always anxious for a double-clickable of the latest version in a timely fashioned, you can't do better than using the officials.
- **Install Using a Package System:** If you're a fan of automated installers like Fink (<http://fink.sourceforge.net/>) or DarwinPorts (<http://www.opendarwin.org/>), there's probably a MySQL package waiting for you (for fink: `fink list mysql`; for DarwinPorts, check the "databases" category). Package systems can usually upgrade everything you've installed in one fell-swoop (fink `selfupdate` and `fink update-all`, for example), and they're one of the easiest ways to *uninstall* a package too (something that most other alternatives have no equivalent).
- **Install an Unofficial Binary:** Various user-created double-clickables exist, most notably ServerLogistics' Complete MySQL (<http://www.serverlogistics.com/mysql.php>). While not "official", they're often "value-added" in some way, and Complete MySQL is no different: it comes with a handy System Preference pane for controlling MySQL (**Figure 3**), a Mac OS X StartupItem (see "Homework Malignments" for

MAC OS X PANTHER

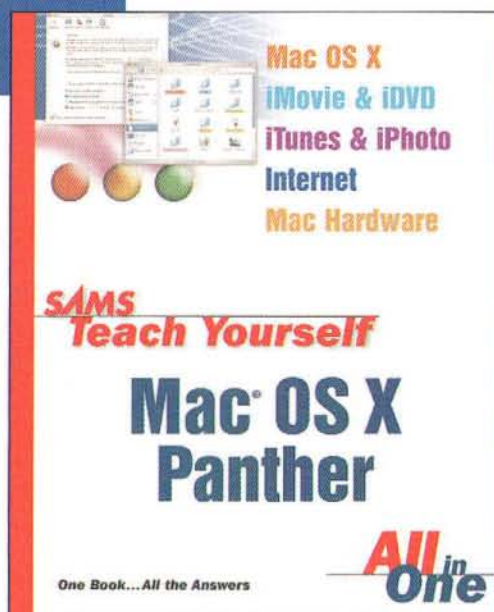
ALL YOU NEED TO KNOW ABOUT PANTHER

Want to know what's new in Panther? Turn to the one book with all the answers.

Sams Teach Yourself Mac OS X Panther All in One

by Robyn Ness and John Ray

ISBN: 0-672-32603-5
\$29.99 US



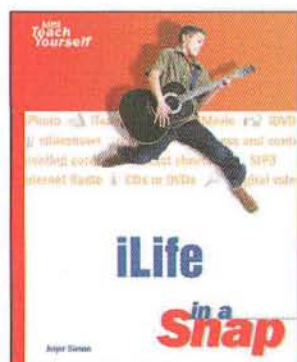
Sams Teach Yourself Mac OS X Panther All in One is designed to teach, in one book, the Mac user how to easily work with the hardware, the operating system, and key applications. Rather than focusing on a single product, the book covers multiple products and technologies together in a logical fashion.

Topics Include:

- Understanding the Mac OS X Panther interface.
- Burning CDs and DVDs with iDVD.
- Playing and organizing MP3s and digital music with iTunes.
- Digital photography with iPhoto.
- Editing digital video with iMovie.

Visit www.amazon.com/samsbooks to download an overview of Panther's new security features

OTHER GREAT SAMS BOOKS ON PANTHER

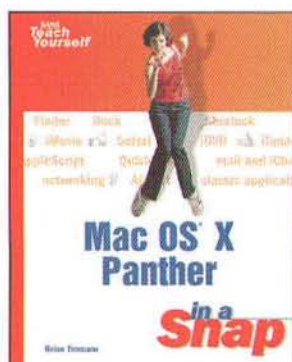


iLife in a Snap

by Jinger Simon

ISBN: 0-672-32577-2 • \$19.99

Available Dec 2003



Mac OS X Panther in a Snap

by Brian Tiemann

ISBN: 0-672-32612-4 • \$19.99 US

Available Dec 2003



Mac OS X Panther Unleashed

by John Ray and William Ray
ISBN: 0-672-32604-3 • \$49.99 US

Available Jan 2004

amazon.com

SAMS

www.sampublishing.com

Available at **DEPOT**

some tips on building your own; the official binary also ships with one), and various drivers for database connectivity with ODBC or JDBC. Unofficial binaries may not be updated as regularly as the official releases, and you run the risk of them not being continued at all (due to lack of interest, impetus, or inspiration).

Which adventure should you embark on? As much as I like the thrills of a shiny new "official" release, the added treasure of ServerLogistics' unofficial binary can save you a couple of steps in your initial exploration. Regardless which path you choose to install MySQL, its continued configuration and administration is the same: you can use the command line to do everything, or nearly everything through GUIs (like CocoaMySQL, phpMyAdmin, etc.) or code (like PHP). We'll cover CocoaMySQL (<http://cocoamysql.sourceforge.net/>) and PHP database coding (<http://www.php.net/mysql>) in our next column and, if there's time, possibly phpMyAdmin (<http://www.phpmyadmin.net/>).

INSTALLING COMPLETE MYSQL

Complete MySQL comes with a rather verbose installation document, but we'll cover the process quickly here. Go ahead and download the latest version (4.0.15 at the time of this writing; the latest official release is 4.0.16). Extract and mount, double-click the MySQL.pkg, agree to everything, and pause for a second when you get to the "Select a Destination" screen.

It continually surprises me how many people don't know about the *File > Show Files* command. Within any installer built using Apple's package system, you can always get a list of the files that are about to be installed simply by choosing that menu item (**Figure 2** shows the file lists for both the official MySQL binary and Complete MySQL). The list can be saved, printed, and used as a roadmap if, and when, you want to uninstall the package (though some creative license will need to be applied to find out where the package's root install directory is, represented as */*).

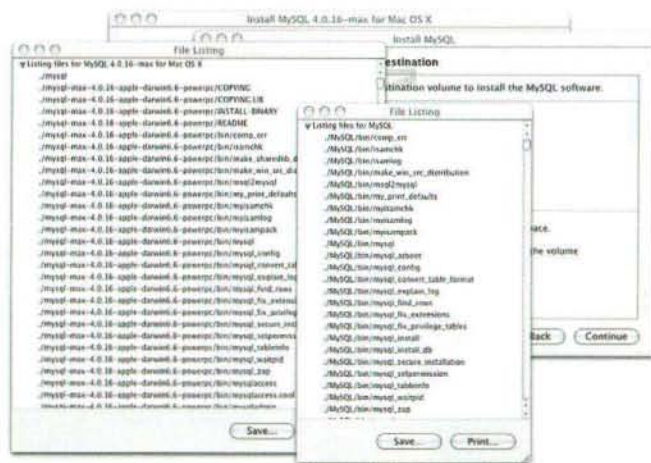


Figure 2: Using "Show Files" gives you a roadmap for uninstalling in the future.

Once the Installer finishes, you should be able to browse to */Library/MySQL* and see a bunch of newly added files. Likewise, you should notice that a MySQL StartupItem has been installed into */Library/StartupItems*. Note that even though MySQL has successfully been installed, it has not been automatically started... there's still some initialization to do.

Next, install the MySQL Preference Pane. Since I alone will be administrating MySQL, I've dragged my copy into */Users/morbus/Library/PreferencePanes/*; if you'd like it accessible to any user on the machine, choose */Library/PreferencePanes/* instead. Once it's in place, open up your System Preferences, and choose the new "MySQL" item (**Figure 3**).



Figure 3: The newly installed MySQL Preference Pane.

To finalize your installation of MySQL, you'll want to first "Initialize" the database, then assign a password to the MySQL root user. MySQL database users have nothing to do with users that you've created within OS X, so don't get them confused. Once MySQL has been initialized, "Start" the server, then "Set Root Password" (**Figure 4**).



Figure 4: Setting MySQL's root password in the MySQL Preference Pane.

We can ensure that MySQL is running a few different ways: by checking the newly created logfiles (at */Library/MySQL/data/computername.local.err*; you'll need to authenticate as an administrative user to do so), or by checking the currently running processes via the shell (*ps auxx*) or Panther's Activity Monitor (**Figure 5**), a much improved replacement for Jaguar's Process Viewer. If you've

got hundreds of processes, use the "Filter" input to narrow in on "mysql".

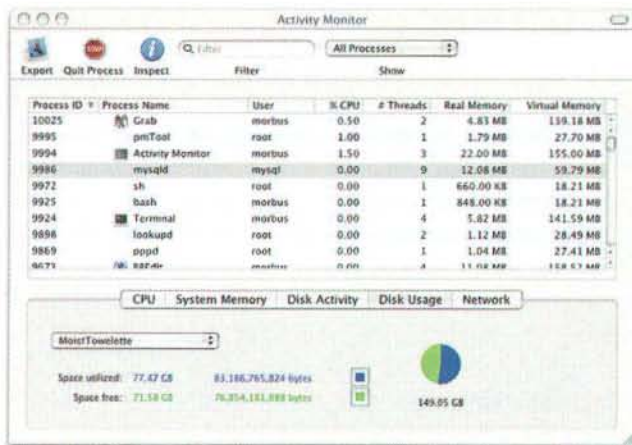


Figure 5: Verify MySQL is running with Panther's Activity Viewer

HOMEWORK MALIGNMENTS

In our next column, we'll look into interacting with our new MySQL database: administrative tasks from the command line, visual interaction with GUI-based tools like CocoaMySQL, inserting and manipulating database records with PHP, and more. We'll really just be scratching the surface on what's possible, but it'll give you enough knowledge to start your own database project, or feel relatively confident about installing someone else's. For now, students may contact the teacher at morbus@disobey.com.

- Who's in the attic?
- In non-Panther versions of OS X, Apple manually installed the optional PEAR, as well as a number of PEAR-released modules, into /System/Library/PHP. In Panther, they probably opted **not** to do this since PEAR became a default PHP extension. Unfortunately, it appears that the PEAR installation under 10.3 is incomplete (witness the "missing System.php" complaint when running /usr/bin/pear). Simply copying an existing System.php from Jaguar won't work, so you'll want to grab the latest from <http://cvs.php.net/cvs.php/php-src/pear/System.php>. Save this file to your Desktop and copy it into place with `sudo cp ~/Desktop/System.php /usr/lib/php`. You should then be able to run PEAR without a problem (list PEAR modules currently installed with `sudo pear list`).
- Every time Mac OS X boots, zillions of things occur... a fair portion of this zillion is controlled by StartupItems, files that say what should run and when. To create your own

StartupItems, take a look at "Creating SystemStarter Startup Item Bundles" (http://www.opensource.apple.com/projects/documentation/howto/html/SystemStarter_HOWTO.html) and "Start Me Up: Writing and Understanding OS X StartupItems" (<http://www.oreillynet.com/pub/a/mac/2003/10/21/startup.html>).

- If you're interested in seeing what the MySQL PreferencePane actually does for you, the installation PDF provided with Complete MySQL has the command line equivalents.

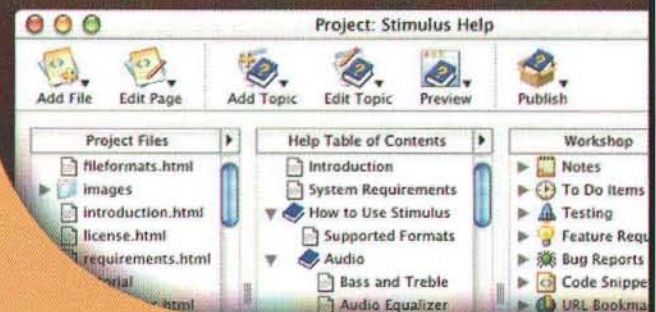


GraphicConverter converts pictures to different formats. Also it contains many useful features for picture manipulation.

See www.lemkesoft.com for more information.

HelpLogic

The Help Authoring Solution for Mac Developers



Easily create help systems for your software applications & web sites from a single source.

Save time with the integrated Workshop, TOC Builder, HTML Editor & Page Templates to quickly generate Apple Help, Web-based Help, UniHelp, PDF & more.

SNEAK PREVIEW
www.ebutterfly.com

electric butterfly

by Joe Zobkiw

RemoteScriptRunner: Remotely execute scripts from any web browser

INTRODUCTION

Lately I've been exploring Java and the many things that can be done with it. I will admit that for years I was turned off by Java due to my initial experiences with it in the Mac OS. When Java was first introduced on the Mac OS it was horribly slow. Most people first saw Java perform in the form of an applet or some other browser-based user interface - unfortunately this was not Java's strong suit. However, with Mac OS X, Jaguar and especially Panther, Apple has made great strides in making Java much more usable on the Mac...which caused me to give it a second look.

Java is not just for user interfaces. In fact, there are so many tributaries flowing off the Java river that it's liable to make your head spin just trying to figure out what Java is actually capable of! In fact, it's probably safe to say that Java is capable of just about any type of software development you are interested in - all you have to do is figure it out or find someone who already has. Java can be used to write clients and servers, access databases, communicate with mobile devices, run appliances, write platform-native code, etc. In this article we look at Java's server side - which arguably is a better choice than many other languages (including C) when writing a server.

RemoteScriptRunner is a proof-of-concept Java application that runs as a daemon process in the background. That is, you won't see any icons in the Dock while RSR is running - like most servers. Although Java can be platform independent, this one implements the ability to execute AppleScript format scripts, so in that regard it is platform specific to any platform that implements AppleScript. However, this mechanism can easily be changed to support other scripting architectures and is left as an exercise to the reader.

RSR is modeled after an article written by David Brown in August 1997 entitled "A Simple, Multithreaded Web Server" which can be found on the Sun Java developer web site at <http://developer.java.sun.com/developer/technicalArticles/Networking/Webserver/>. I recommend you look at this article for details on the server as I will not delve into the details of that here. In fact, this article takes more of a "here's what I learned"

approach. The AppleScript portion of this code is modeled on a code example by Scott D.W. Rankin and is available at <http://macdevcenter.com>.

20,000 FEET

At 20,000 feet, RSR functions as follows. You double-click the compiled Java application, usually in the form of a JAR file. Although you won't see it in the Dock, suffice it to say that this starts the server running and waiting for client connections on port 8080, or any port you specify. You can check to see if the server is running by typing `ps ax | grep java` in Terminal. If the server is running you should see a line in the result that looks something like `460 ?? S 0:00.71 java -jar /Users/zobkiw/RSR/RemoteScriptRunner.jar`. At this point the server is active yet essentially idle as it awaits a connection from a web browser client.

Next, a user launches their web browser and accesses the server as they would any other HTTP URL. In our case, since we are testing things locally, we use `http://localhost:8080/` but localhost can be 127.0.0.1 or any other valid IP address or server/domain name. The server accepts the HTTP connection, sees that the request is in GET format and returns the HTML necessary to display a form to the user. This form contains an editable text area, a submit button and a button to quit the server. The editable text area is used to type your AppleScript.

Once an AppleScript is entered, you press the Submit button to POST the form and data to the server. This time when the browser connects to the server, the server accepts the connection and sees that the request is in POST format and parses the data

MacTech®
M A G A Z I N E

Get MacTech delivered to your door at a price **FAR BELOW**
the newsstand price. And, it's **RISK FREE!**

Subscribe Today!
www.mactech.com

Joe Zobkiw is the author of *Mac OS X Advanced Development Techniques* and President of TripleSoft Inc., a software development and consulting company in Raleigh, NC. He can be reached at zobkiw@triplesoft.com.

in the form — most importantly, the AppleScript. The server attempts to execute the script using the necessary AppleScript-related Java classes and returns the results as HTML to the client.

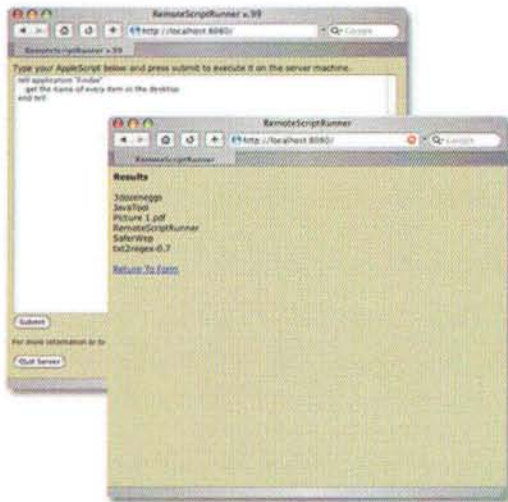


Figure 1 - RemoteScriptRunner in action

In Figure 1 we show the two "pages" created by the server. In the first page (behind) we have filled out the form to include an AppleScript that tells the Finder to get the name of every item in the desktop. After submitting this script, we see the results in the second page (front). Note that the results are simply a list of items that happened to be on my desktop at the time.

10,000 FEET

Zooming down to 10,000 feet, let's take a look at the development environment and source code. Java development tools can take you in many different directions - all have their advantages and disadvantages for any particular project. There is Project Builder for pre-10.3 users, Xcode for 10.3 users and beyond, Eclipse for anyone on just about any platform, and then BBEdit and the command line. Although I've written Java code using each of these options, for this project I chose BBEdit and the command line. When there isn't a whole lot of code to write I can do things just as easily and usually faster by using these stand-by tools.

So, in this project I created my RemoteScriptRunner.java file in BBEdit as well as the manifest file, RemoteScriptRunner.mf. Then I use Terminal to compile, run and build the JAR file. The command line to compile is `javac -classpath /System/Library/Java:. RemoteScriptRunner.java`. The command line to run is `java -classpath /System/Library/Java:. RemoteScriptRunner`. The command line to build the JAR file is `jar cmf RemoteScriptRunner.mf RemoteScriptRunner.jar *.class`. The manifest file is a text file that contains two lines, as follows:

```
Main-Class: RemoteScriptRunner
Class-Path: /System/Library/Java/
```

Because the easily accessible Brown article mentioned earlier does such a good job at explaining the multithreaded nature of the server, to which I made few changes, I won't go into the details of that here. This article will primarily discuss the `handleClient` method of the Worker class that is called after a connection is accepted. However, let's quickly discuss what leads up to the `handleClient` method being called.

First, the main program loads all program settings and creates a series of Worker objects as Threads. Because it's less "expensive" to create a few of these up-front, we do it at program initialization rather than when a connection is actually established. These Worker objects are stored in a Vector and are available to handle connections as they are accepted. The server then establishes itself and loops forever, waiting for a connection. When a connection is established, the first Worker object not already busy is pulled from the Vector and passed the Socket that accepted the connection. At this point the Worker object, which was in a wait state, is notified to wake up and begin its work, ultimately calling its `handleClient` method.

In `handleClient` the first thing you want to do is create a `PrintWriter` on the socket's output stream for writing and a `BufferedReader` on the socket's input stream for reading. We also set the timeout so we don't hang the machine in the case where the socket is left open but there is nothing left to read.

```
// Create a reader and writer
PrintWriter pw = new
```

TelnetLauncher

Bookmark and Launch your
Telnet and SSH sessions

Featuring:

- > Password storage
- > Auto minimize
- > Set text font and size
- > Specify terminal colors
- > Port forwarding
- > Terminal emulation

This and much more available from...

piDog Software

SimpleKeys - piPop - DockSwap - ScreenShot Plus

www.pidog.com
Info@pidog.com


```

PrintWriter(clientSocket.getOutputStream(). true);
BufferedReader br = new BufferedReader(new
InputStreamReader(clientSocket.getInputStream()));

// We will only block in read for this many milliseconds before we fail with
// java.io.InterruptedIOException, at which point we will abandon the connection
clientSocket.setSoTimeout(RemoteScriptRunner.timeout);
clientSocket.setTcpNoDelay(true);

```

Depending on what the purpose of your server is, you can very easily loop calling the `readLine` method of the `BufferedReader` repeatedly until no more data is available. This will get you (most) everything coming from the client.

```

// Read the bulk of the data from the input by line
String s;
while ((s = br.readLine()) != null && (s.length() != 0))
    System.out.println("> " + s);

```

We do just this with the addition that we also look for specific information coming from the client. In the HTTP protocol the server receives all sorts of information from the client when a connection is open. A part of this information includes the type of request. Although we can just as easily look for specific codes and values within the URL or embedded in the data, in this implementation we look for GET and POST requests specifically and base our response on that. We make use of the `startsWith` method of the `String` class for this purpose. If the string starts with "GET" then we respond by sending back the HTML containing the form. If the string starts with "POST" then we know that the form is being submitted and we extract the AppleScript from it and attempt to execute it, returning its result.

One thing to note about the `readLine` method used above is that in the case of a POST, `readLine` will not read the POSTed form data. The problem is that the form data does not end in a newline character, so `readLine` essentially ignores it. Given that, in the case of a POST, we have to finish reading the data character by character using the `read` method of the `BufferedReader`. As we read each character, we build a string containing all of the data.

```

// Read the rest of the available data and create a string of it
s = "";
while (br.ready() && ((ch = br.read()) != -1))
    s += (char)ch;

```

Once you have the string containing the form data there are a few things to do to it before you use it. First we trim the string using the `trim` method of the `String` class. Next we decode the string using the `decode` method of the `URLDecoder` class, passing "UTF-8" as the decoding scheme. Then, using the `StringTokenizer` class we split the string by "&" to extract the name and value pairs. Once we have each pair we use the `StringTokenizer` class once again to split the string by '='. This gives us the value of any particular field from the form. The `getFormVariableValue` method shows the use of the `StringTokenizer` class. It assumes a string passed in such as `script=beep 3&something=this&somethingelse=that`.

```

// Get a form variable value from a list of variable name and data pairs

```

```

String getFormVariableValue(String variables, String name)
{
    // Set up our first tokenizer and variables
    StringTokenizer st1 = new StringTokenizer(variables, "&");
    String s1 = "";
    String n1 = name.toLowerCase() + "=";

    // If the given name is not even in the variables then exit immediately
    if (variables.toLowerCase().indexOf(n1) == -1)
        return null;

    // Search for first token as a name and data pair (ie: variable=data)
    while (!s1.startsWith(n1) || s1.length() == 0) {
        s1 = st1.nextToken();
        System.out.println("s1=" + s1);
    }

    // Now that we have the first token, we can split it into the name and data specifics
    StringTokenizer st2 = new StringTokenizer(s1, "=");
    String s2 = "";
    String n2 = name.toLowerCase();
    while (s2.startsWith(n2) || s2.length() == 0) {
        s2 = st2.nextToken();
        System.out.println("s2=" + s2);
    }

    return s2;
}

```

At this point, by extracting the value of the "script" form variable we finally have the raw AppleScript to execute. We first create a new `NSAppleScript` object by passing in the script. We then create an `NSMutableDictionary` object to hold any errors during execution. Sending the `NSAppleScript` object the `execute` message causes the script to execute and return results in an `NSAppleEventDescriptor` object. The results in that object can then be extracted and displayed.

```

NSAppleScript myScript = new NSAppleScript(s);

// This dictionary holds any errors that are encountered during script execution
NSMutableDictionary errors = new NSMutableDictionary();

// Execute the script!
NSAppleEventDescriptor results = myScript.execute(errors);

// If multiple items in the result we use this to display results
int numberOfItems = (results == null) ? 0 :
results.numberOfItems();
for (int i = 1; i <= numberOfItems; i++) {
    NSAppleEventDescriptor subDescriptor =
results.descriptorAtIndex(i);
    System.out.println(subDescriptor.stringValue());
}

// If only one item in the result we can use this
if (numberOfItems == 0) {
    String resultString = (results == null) ? "" :
results.stringValue();
    System.out.println(resultString);
}

```

CONCLUSION

As mentioned, RSR is a proof-of-concept for a larger project of mine. There are many ways to improve this code and even more possible features to add. There are also other ways to remotely invoke scripts, but this was a fun project to put together that works reliably. In closing, I hope this convinces you to give Java a second chance — you just might like it!

Technological.

The logical way to connect your
news and information to the people who count.

At PR Newswire, our approach to distributing technology news is simple: target the people with a real interest in your news – journalists, investors, analysts and consumers – and deliver your message to them in the way they prefer.

PR Newswire has earned the respect of the media and financial community by consistently delivering accurate, credible, reliable news and information directly to their desktops. We also offer a full range of customized solutions to help get your message across with photos, VNRs, Webcasting, and more.

Make sure your news is seen in all the right places.

Call PR Newswire at 888-776-0942 or visit us at www.prnewswire.com.

We tell your story to the world.™



PR Newswire
United Business Media

By John C. Welch

New Networking Tricks in Panther

A quick look at some new tricks, good and bad, in Panther

WHAT'S UP DOC?

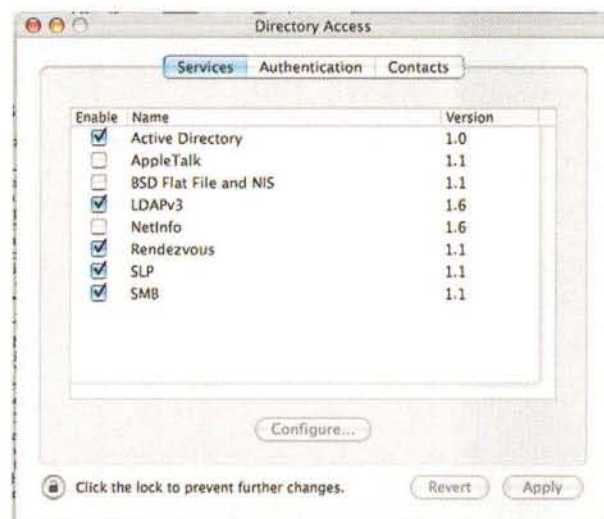
So, once again, we have a major release of Mac OS X upon us, and while everyone else is ooh-ing and ah-ing things like Expose and vertical toolbars, I, MacTech's intrepid IT Geek, am plumbing the networking changes to Panther. Well, okay, so most of the UI stuff is not that exciting to me. Expose is cool I suppose, but I had my own methods of dealing with this that don't require Quartz Extreme, or yet another set of key equivalents. (Lately I've been having flashbacks to WordPerfect 5.1, and the dorky key-command templates we all needed with it.)

But Panther is a big change, and I daresay that if you're a networking type, (I'll assume you are, since you're reading a column called "Patch Panel"), Panther has a lot of stuff for you. Now, I'm not going to be able to look at everything in this column. To do so would mean the "John C. Welch" issue of MacTech, and while that's a great ego stroke, there's a hard limit to how much of me anyone should have to take in a month. Instead, I'll take a look at a key element of Panther's networking subsystem.

Active Directory Integration

This is one of the most improved areas of Panther, and the changes are a long time in coming. Personal opinions of Microsoft notwithstanding, it's just sensible for Mac OS X to play nice with Active Directory. In this area, the biggest new toy is the Active Directory, (AD) plugin for Directory Services. This allows you to make your Mac a member of an Active Directory domain, and be able to play almost as nicely as a Windows box. (I say "almost" because there's a huge part of AD that requires windows, such as MSI, certain group policies, ACLs, etc.) This is quite different from the way you did this in Jaguar, which used the LDAP connector to talk to AD, and could sometimes require modifying the AD schema to work right. (To be fair, there's

nothing wrong with using LDAP. It's how AD does a lot of its work, and even with the plugin, if you want certain things stored in AD, you're still going to need to modify the schema. But the plugin minimizes some of this.) As well, enabling automatic Kerberos authentication at login with Jaguar was somewhat tricky, and not for the faint of heart. You could use ADmitMac, from Thursby Systems with Jaguar, and get the same level of integration as the Panther plugin enables, along with some extras, such as better use of Windows shares, (no .DS_Store booger files littered everywhere), and support for NT 4 Domains, (Panther's plugin is AD only.) So, if you need to deal with NT 4 domains, or need to integrate Jaguar with AD, ADmitMac is a great solution, albeit not free. But then again, neither is Panther. In any case, with Panther, AD just got a lot easier.



Directory Access in Panther

So, as we see above in the Directory Access application, AD now has its own entry. Click on Active Directory, hit "Configure..." and you get the following screen:

John Welch <jwelch@provar.com> is a Technical Strategist for Provar, (<http://www.provar.com/>) and the Chief Know-It-All for TackyShirt, (<http://www.tackyshirt.com/>). He has over fifteen years of experience at making Macs, and other computers work. John specializes in figuring out ways to make the Mac do what nobody thinks it can, showing that the Mac is a superior administrative platform, and teaching others how to use it in interesting, if sometimes frightening ways. He also does things that don't involve computers on occasion, or at least that's the rumor.



Active Directory Plugin Configuration

To add, or bind a computer to an Active Directory domain, you enter in the forest name, the domain name, and the computer name. If you don't have a separate domain, then use the forest name. Click on the "Bind..." button. (It says "Unbind..." here because my laptop is already bound to a domain.), enter in your Mac OS X admin password, and the userid and password of a user that is able to add machines to an AD domain, and you're set. Note that the userid and password for the AD domain does not have to be the logins for local users on that machine. In my case, they're completely different. If there are no errors, then your machine is now a part of an AD domain. There are a few options that can make your life easier here. "Cache last user logon for offline operations" is very handy for laptops, so that you can log onto your machine and get work done, even when you're off the network. If you have a large AD forest, the "Authenticate in multiple domains" can make your life easier. "Prefer this domain server:" allows you to specify what domain server to authenticate to when available. Unless you have a specific need for this, leave it unchecked. (If you have to ask if you need this, the answer is probably "no".) "Map UID to attribute:" allows you to map the unique User ID to a specific AD attribute instead of letting AD handle this. Again, if you aren't sure you need to do this, leave it alone. "Allow administration by:" lets you give admin rights to users in certain domain groups. By default, the domain admins and enterprise admins groups are used if this option is enabled, and you can add others if you like. This allows AD administrators to have administrator rights on a Mac OS X machine without having to create local accounts for them.

However, if you have to set up the plugin on multiple machines, using the UI tools can get a bit tedious. They still

work, but automating the Directory Access application is fairly tedious. Luckily, you can completely set up the AD plugin via the command line, and the dsconfigad application. The man page for dsconfigad is pretty complete, and has some nice examples. So, to add a machine to a domain, the command line would look like the example in the man page:

```
dsconfigad -a ThisComputer -u "administrator" -ou
"CN=Computers,OU=Engineering,DC=ads,DC=demo,DC=com" -forest
ads.demo.com -domain domain.ads.apple.com
```

The man page gives clear examples of using dsconfigad to set up all the different features of the plugin. Including a command line configuration option makes the plugin much easier to use with other management tools, even if your management console is running a different flavor of Unix, or even Windows. Cross-platform automation in the IT space is *a good thing*.

Authentication and Contacts setup

By default, Mac OS X will search through the local authentication domains first, then any external directories. If you have multiple authentication directories, or you want to force a specific order, then you can create custom authentication paths as in the image below:

MacTech®

M A G A Z I N E

*The source for developers and the
technical market since 1984.*

CRASH PROOF

- No installer needed
- No download times
- Readable in your favorite reading locations
- No drain on your system's resources
- The best reader user interface mechanism available Monthly and delivered painlessly to your doorstep.

Authored by a number of industry experts
working in the real world

Get it today **RISK FREE** at <http://www.mactech.com>



Setting custom authentication paths

This tells Directory Services where to look, and in what order when performing authentication operations on a given machine. Now, there's another thing that we use directories for, namely as distributed address books. If we take a look at the "Contacts" tab in directory services, we see that it looks much the same as the Authentication tab, and you can set up custom search paths there as well, like in this image:



Setting custom Contacts paths

If you do this in Panther with Active Directory, you get one immediate bonus. Address Book, and therefore Mail can now use Active Directory's Global Address List, or GAL, to look for email addresses when sending mail.

Once you have this set up, how's it work? Well, pretty darn well so far. I can log into my laptop using my AD login identity, with no local account creation. My home directory is created, and

I have access to all my Mac applications. When I connect to shares on the Windows network that I have access to, I don't have to supply additional credentials for them, they just work. So the single signon aspects of Active Directory work with Panther as well. This is due to the other part of the AD plugin's magic, namely it's Kerberos support. When I log into my AD domain, since AD and Panther both heavily use Kerberos, I automatically get my Kerberos tickets. So when I attempt to use AD services, like access to network file shares, I don't have to re-enter my user information. One signon does it all, thanks to Kerberos.

The only real problem I ran into was a momentary problem with DNS. Like a lot of network services in Mac OS X, the AD plugin makes heavy use of reverse DNS lookups to get information on the AD domain so that it can interoperate correctly. When I first tried to bind to the domain, I kept getting reverse DNS errors. Nothing seemed to be wrong, and by the next morning everything was working fine, and I could bind with the domain, so I'm not really sure what went wrong there, or what got fixed, since nothing was changed on the AD side.

This is a major benefit to Apple and Mac OS X in almost every market they compete in. Regardless of your opinion of Microsoft, Active Directory is one of the most popular directory systems on the market, and with good reason. It's flexible, fairly secure, (As a product. While Windows tends to have a lot of security holes, AD has been pretty clean here.), and had excellent management tools. It's very dominant in the enterprise, and is gaining ground in both the higher ed and k-12 markets. Integrating well with AD is critical for Apple to go from a reluctantly accepted platform to an accepted alternative to Windows on the desktop and Linux in the server room.

I know that in my case, the ease of setup of the plugin, and the functionality it provides is going to make my Macs a much more accepted part of the network. This doesn't mean that we automatically start buying Macs by the truckload, but in the future, if I bring up Mac OS X as a solution to a problem, there won't be the automatic "Macs can't integrate with AD" dismissal.

There are still a few things that need to be done on the integration side, such as creating a Microsoft Maintenance Console, (MMC) snap-in for Mac OS X, so that you can properly manage Macs with the Windows AD administration tools. Giving Windows administrators a way to use Group Policies with Macs would be another good idea too. Since most Mac OS X applications don't use a resource fork, it has more flexibility with installation sources than Mac OS 9 did, so there is at least a theoretical potential for MSI integration that I would like to see explored a little more. However, for a first implementation, the plugin works quite well.

CONCLUSION

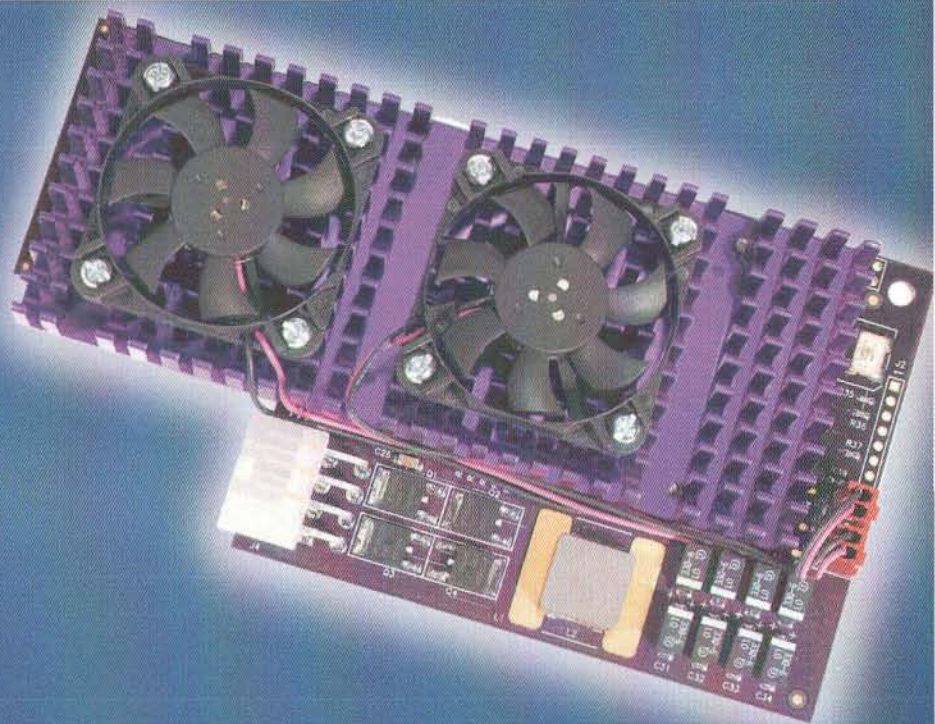
Obviously Panther contains far more networking improvements than just an Active Directory plugin, but the plugin is a major new feature that will help Apple be thought of as a much better player in the enterprise space. No matter how you look at it, this can only be thought of as good for Apple and the Mac community.

SONNET™

SIMPLY FAST™

Upgrade Cards

Put a G4 in your iMac,
Beige, Blue and White,
or other Mac.



Yea ... it feels like that.

www.devdepot.com/upgrades

DEV DEPOT®

www.devdepot.com

877-DEPOT-NOW



Toll Free: 877-337-6866

Outside US/Canada: 805-494-9797

Fax: 805-494-9798 • orders@devdepot.com

PO Box 5200 • Westlake Village, CA 91359-5200

By Dave Wooldridge

Driving Traffic to Your Web Site

Online Publicity for Your Software

Last month, we explored how to turn your web site into a powerful marketing tool for your software. While those techniques will help improve your site's effectiveness, you still face the task of getting consumers to visit your site. Some developers are under the false impression that if they launch a web site, then a flood of business will automatically come their way. In truth, the World Wide Web gives people access to any connected web site, but in order for them to find your site, they first need to know that it exists. A web site is no different than an isolated retail store. Sure, you may get some occasional foot traffic from people who accidentally stumble upon your storefront while on their way to another destination, but you're certainly not going to break any sales records relying on that method alone.

Getting your site listed in the major search engines (such as Google and Yahoo!) is definitely a move in the right direction, but is that enough? If a consumer searches for the right related keyword, your site may be listed in the results, but will it rank in the top ten? Will it even appear on the first page of search results? Improving your ranking status in the search engines is a science in itself (which we will examine later in this article), but luckily for software developers, it's not the only avenue for reaching potential customers online.

TARGETING YOUR MESSAGE

Not everyone is going to be interested in your software. As we've discussed in previous issues, your objective is to focus your marketing efforts on the select group of people who you consider to be potential customers. This is your product's target market. These people would have a need or interest in your software product. Your next task is to figure out how to reach this target market online. How do they find software on the Web? What sites do they visit? Once you figure out this important equation, you'll have a clear picture of how and where to best spend your Internet-based publicity efforts and advertising budget.

Why is this important? Knowing your audience allows you to tailor your marketing message to address their specific needs. Plus, this targeted campaign enables you to speak directly to

potential customers, so that your time and money are not wasted promoting your product to the wrong audience.

Let's use our fictional software product, CodeQuiver, as an example. CodeQuiver is a handy Mac utility for storing and organizing code snippets. Since the new version 1.5 would primarily benefit software developers and web designers, our goal should be to reach as much of this target market as possible. With this in mind, we would want to send our CodeQuiver 1.5 press release to any site or mailing list that announces Mac-related developer news such as MacTech.com, MacNN.com, MacInTouch.com, MacMinute.com, Apple Developer Connection News (<http://developer.apple.com/devnews/>) and the dozens of other popular Mac sites.

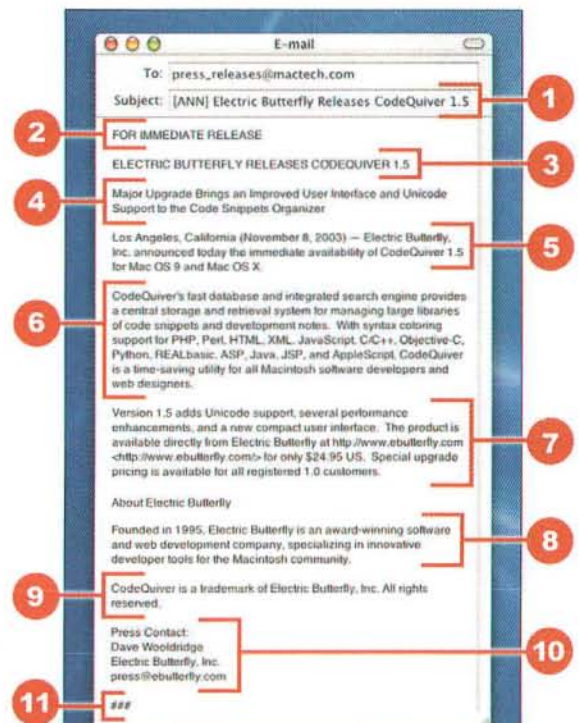


Figure 1. A fictional e-mail press release for CodeQuiver, with items 1–11 representing the essential elements.

Dave Wooldridge is the founder of Electric Butterfly (www.ebutterfly.com), the web design and software company responsible for Stimulus, HelpLogic, UniHelp, and the popular developer site, RBGarage.com.

ANATOMY OF AN EMAIL PRESS RELEASE

With e-mail being a convenient delivery mechanism, most magazines and web sites prefer to receive press releases via e-mail. Usually, they have dedicated e-mail addresses for collecting press releases, although finding them often requires a little online research. If these e-mail addresses are not listed in the obvious places, then you can usually find the right contact information and submission instructions on either their staff pages or writers kit pages. MacTech's e-mail address for press release submissions is press_releases@mactech.com.

Even though e-mail is often viewed as a very informal medium, there are rules on how to properly write and format e-mail press releases. **Figure 1** represents a typical software press release, formatted for e-mail delivery. Since many sites have automated scripts that convert e-mail press releases into web pages or RSS/XML feeds, be sure to always use Plain Text with standard ASCII characters. Never e-mail a press release to a web site in Rich Text or HTML format. Avoid using non-standard characters such as long dashes and curly quotation marks, which may get incorrectly converted into garbled text, making your press release look unprofessional (not to mention unreadable) when viewed on these news sites. Always e-mail yourself a test copy of your press release so that you can proof read it for any odd character conversions or typos (and don't forget to run it through a spell checker).

1 – E-mail Subject. The subject line is usually the same as your press release's headline, with the exception that the subject line is not all capital letters (see **Figure 1, Item 1**). Some newsgroups and mailing lists require press releases and announcements to use an indicator such as [ANN], so check the submission instructions of each site or group before e-mailing your press release.

2 – First Line. The words "FOR IMMEDIATE RELEASE" in all capital letters should be the first line of your press release (see **Figure 1, Item 2**).

3 – Headline. When viewed in print, the press release headline is usually a bolded, larger font, so capitalizing the letters is often viewed as optional. Plain Text e-mails do not feature bold styles or variable font sizes, so headlines are emphasized in e-mail with all capital letters (see **Figure 1, Item 3**). This headline is usually the title that appears on the news sites as a text link to your press release (see **Figure 2**), so take special care when writing your headline (and matching subject line) to ensure that it includes your company name, product name, version, and the key words that describe your announcement. Software companies that produce shrink-wrapped products often use the word "ships" (as in "Electric Butterfly Ships CodeQuiver 1.5"), instead of the default verb "releases" which encompasses all software, including download-only shareware.

Macintosh Software News	Feature Story
<ul style="list-style-type: none">• Nov 8, 2003 - Maximum Money 2.0 is Now Shipping• Nov 8, 2003 - Electric Butterfly Releases CodeQuiver 1.5• Nov 8, 2003 - Blast2Bits 1.4b1 Now Available• Nov 8, 2003 - Text Editor 1.0 for Mac	The highly anticipated release of Mac Panther includes more than what you're making it a

Figure 2. The wording of your Headline and Email Subject line are very important since news sites often post your press release using that title phrase as the listed hyperlink text.

4 – Secondary Headline. Some press releases include a secondary caption beneath the main headline that provides an expanded, one-sentence synopsis of the announcement (see **Figure 1, Item 4**). This line is optional, but is strongly recommended for online viewing. The first ten to fifteen words of a press release are usually considered the most important. If a reader does not find anything of interest in the first paragraph, then they are typically not inclined to read the rest of the release. Viewing paragraphs on a computer monitor is even more tedious, so motivate people to read the rest of your press release by enticing them with a summarized caption.

5 – Lead Paragraph. The first phrase should list your company's city and state (or country when located outside of the U.S.), followed by the date. The formatting of this phrase varies – some people display the city in all capital letters, while others place the date within parentheses. This paragraph is usually quite brief, consisting of only a few sentences, but it should communicate the five essential W's of your message: who, when, where, what and why (see **Figure 1, Item 5**). For a product announcement, this paragraph usually describes how your company is announcing the immediate availability of your new software version and the related platforms it supports.

6 – Main Text. The next one to three paragraphs after the lead paragraph reveal product details and key features that relate to this release (see **Figure 1, Item 6**). Even though your description should effectively promote the benefits of your software, try to avoid hyped marketing language that makes your announcement sound like an infomercial. Subjective adjectives like "ultimate" do nothing but consume valuable space. With hundreds of new software announcements vying for attention every day, a good press release is concise and to the point, fitting on one or two printed pages (but never exceeding two).

7 – Availability and Pricing. This paragraph should reiterate any specific new product features or updates while stating exactly how and where the product can be purchased. This includes your web site URL, sales phone number (if applicable), pricing and any upgrade offers that are available for existing customers (see **Figure 1, Item 7**). If your software requires any unique system requirements, you should also include that information in this paragraph. If you are sending your press release to a mailing list, some e-mail programs only display URLs as clickable hyperlinks if they are surrounded with "<" and ">" tags. For example, write the URL "<http://www.codequiver.com>" as "<<http://www.codequiver.com>>" for best

results in e-mail viewers. Consumers can't buy your software if they don't know where to find it, and yet the accidental omission of the software site URL is the most common mistake shareware developers make when writing press releases.

8 – About Your Company. After the main body of your announcement, you should include a brief paragraph about your company (see **Figure 1, Item 8**). How long have you been in business? Does your company specialize in a specific kind of software? Are you well known for a particular product? Have you won any awards? This not only familiarizes readers and news editors with your company name, but if your company has achieved any kind of industry recognition, it lends credibility to your software announcement. It's important to draw attention to the notion of credibility because it has a powerful effect on consumer perception of your software. Claiming that you're "a one-person virtual company (if you don't count the three cats who occupy your home office) who enjoys playing Unreal Tournament in your spare time" is probably not cute or endearing to the general public. It merely diminishes your credibility, making your company seem like a small, garage operation.

9 – Legal Disclaimer. This line is optional, but recommended. You are announcing your product to the world, so protecting yourself is never a bad idea. State not only your own trademarks (see **Figure 1, Item 9**), but also safeguard yourself by recognizing the third-party trademarks of unaffiliated products that you reference in your press release. Many Mac developers list Apple's trademarks in their press releases. Java developers usually include a reference to Sun Microsystems's Java trademarks.

10 – Contact Information. Hopefully, your press release will be read not only by consumers, but also news editors and software reviewers. If these professionals decide to write an article or review on your software and require additional information, make it very easy for them to contact you. Since your press release is posted on news sites as well, you should think carefully about what information you include. If you're a shareware developer working at home, you may not want to list your phone number (unless you enjoy being awakened at 3:00 AM by an unknown caller from Germany with a software question). Simply include your name and a direct e-mail address (see **Figure 1, Item 10**). If your company has a dedicated press contact, then include that person's job title as well.

11 – End Indicator. Placing three pound symbols "# # #" on a separate line is a standard indicator marking the end of your press release.

Even though the focus of this article is on driving traffic to your web site, do not abuse the power of press releases. Only send out an announcement when you have something newsworthy to say, such as a new product release, a major version update, a strategic alliance or partnership deal, etc. The cool new buttons you designed for your home page are not newsworthy. If you flood news editors' inboxes with frivolous press releases, they will start treating you like the boy who cried wolf, ignoring everything with your company name on it. Then

you will find getting their attention quite difficult when you finally do have something important to announce.

AN ONLINE PRESS ROOM

Put your press releases to work on your own web site. Create a dedicated "Press Room" or "News" page where you can list links to all your press releases in both HTML and PDF formats. These additional HTML pages (chock full of vital keywords) will get indexed by search engines and should improve your search result rankings, which in turn helps media professionals and potential software customers discover your web site. Providing alternative PDF versions allows journalists to download the press releases for offline reference. If you're running Mac OS X, you can produce PDF versions of your press releases from literally any application (such as Microsoft Word or TextEdit) by choosing "Print" and then clicking the "Save as PDF" button.

SOFTWARE WEB DIRECTORIES

While some dedicated software sites like MacUpdate.com still accept e-mail press releases for listing submissions, most of these online software directories require you to manually complete a web form for each new product you want listed. Since a large percentage of Mac users frequently visit sites like VersionTracker.com, MacUpdate.com, MacOSXApps.com, Download.com, Tucows.com, Jumbo.com and Apple's Macintosh Products Guide (<http://guide.apple.com>) to find software solutions and updates, it's in your best interest to take the time to list your product with each and every one of these popular sites.

While the information that these web forms require is roughly the same as what you included in your press release, one key difference is the addition of a short description field. This field usually holds a short phrase that describes your product in ten words or less. This phrase will accompany your product name and version number when line listed on the software directories' home page or category pages (see **Figure 3**). You may have a cool product, but the only way site visitors will ever know this is if they click on your product listing for more information. Since most consumers quickly scan these lists for interesting software, your short description just became the most important element of your software listing because it needs to entice a user to click on your product link.

Today's Mac OS X Software Updates		Size	License
Maximum Money 2.0 - Accounting Application.		2.4M	Demo
CodeQuiver 1.5 - Time-Saving Code Snippets Organizer for Developers		1.2M	Shareware
Blast2Bits 1.4b1 - 3D Shooter Game		3.5M	Beta
Text Editor 1.0 - Yet Another Text Editor for Mac		814k	Shareware

Figure 3. Your product listing's short, one-sentence description on software web directories like VersionTracker.com and MacUpdate.com needs to motivate users to click the link.

Think of it as the "single sentence sell." If you built an accounting program, don't just list it with a short description of

"Accounting Application for Mac." Since there are dozens of accounting applications available for the Mac platform, that generic description simply does not do your product justice. What makes your product unique from the rest? This phrase should include the keywords and features that will attract your target audience in ten words or less. For example, our fictional product, CodeQuiver, is a code snippets organizer for developers. Why would developers want to use it? Because it's designed to save them precious time and effort. Using the phrase "Time-Saving Code Snippets Organizer for Developers" accomplishes this goal in only seven words.

SEARCH ENGINE STRATEGIES

Many of the major search engines have resorted to paid listings to generate revenue. While paid listings will guarantee placement and drive targeted traffic to your web site, budget-conscious developers can still find search engines that accept free submissions. Of all the major search engines, Yahoo.com and Google.com are arguably the most important to your software business. Other popular search engines include LookSmart.com, Search.AOL.com, MSN.com, AltaVista.com, Ask.com (Ask Jeeves), Lycos.com, Excite.com, AllTheWeb.com, About.com, DMOZ.org (Open Directory Project), etc.

The list of sites may seem a little overwhelming, but the truth is that many of them utilize the same search technology under the hood, so if you're listed in one of them, chances are you're listed in many of the others as well. Overture and Inktomi

(which are owned by Yahoo!) power the search results for dozens of sites such as AltaVista.com, AllTheWeb.com, About.com and MSN.com. Many sites such as Search.AOL.com also provide additional content from DMOZ.org (Open Directory Project). Yahoo.com and Search.AOL.com both utilize Google's search results, and hundreds of affiliate sites display Google's sponsored AdWords links (<http://adwords.google.com>).

To navigate through the confusing landscape of search engine marketing, it's best to devise two strategies: one for purchasing paid placement and one for free submissions. While it might seem like a no-brainer to focus only on the free submissions, none of the search engines guarantee placement, and even if your free site listing is accepted, it could take weeks (or even months) for the listing to appear online. Paid placement not only guarantees your listing (with fast, priority processing), but some sites will not accept free submissions from commercial entities (such as a software company), so paid placement is often your only option.

For those developers with limited marketing budgets, a good starting point is to submit your site URL to as many search engines as possible that accept free submissions. There are several software tools available for submitting your site to dozens of search engines from a single form, but the major search sites have become very savvy at recognizing spam and automated submissions and often reject those listings. Manually submitting your site to each individual search engine requires a lot of time and patience, but it often yields the best results and

Expand your LAN

TrangoLINK-10™ Outdoor Wireless Ethernet Bridge

- ◆ 10 Mbps up to 40 Miles
- ◆ Easy to Install
- ◆ Easy to Manage
- ◆ Secure



Expand your existing LAN in hours! The TrangoLINK-10 point-to-point outdoor wireless Ethernet bridge offers secure, high-speed transmission in the license exempt 5.8 GHz and 5.3 GHz bands. Ideal for building-to-building connectivity, the TrangoLINK-10 can be integrated into existing networks for a low-cost, scalable LAN expansion solution.



www.trangobroadband.com

Phone: 858.653.3900

Email: sales@trangobroadband.com

Fixed Wireless that Works!

fine-tuned control over selected keywords and directory categories. Most search engines such as Google provide links named "Add a URL" or "Suggest a Site." Follow their posted instructions and see what kind of results you get in the months to come. If you're not happy with your search ranking in Google and other search sites, then you can always invest in paid placement campaigns.

Yahoo! accepts free submissions for non-commercial sites, but all commercial sites are now required to sign up for Yahoo! Express for US\$299 per year. Yahoo! can still decline your site listing, but your Express submission is quickly evaluated within seven working days. A frequently overlooked benefit of paid placement in human-compiled web directories like Yahoo! and LookSmart is that it often influences your search ranking in other search sites like Google. If you do purchase paid placement on various search engines, make sure you read the fine print. Some plans require you to pay per click on top of the initial service fee, and although the cost per click is usually very low, it definitely can add up quickly. If your listing garners a lot of click-throughs, your "pay for play" campaign may prove to be both successful and expensive.

There are companies for hire that specialize in search engine placement and optimization. At one time or another, we've all received spam e-mails with special offers that guarantee "top ten" placement in hundreds of search engines for only US\$39.95 or less per month. If you decide to hire one of these firms, be very careful who you choose. There are many reputable experts who produce stellar results, but there are even more scam artists who will take your money and run. Do your homework and research the milestones they've achieved for other clients before signing on the dotted line.

OPTIMIZING YOUR SITE FOR SEARCH ENGINES

Internet marketing gurus often talk about foolproof web page optimization solutions for improving your site's rankings in the search engines, but the plain truth is that there is no one "magic bullet" technique that works across the board. All the search engines have their own unique indexing and ranking algorithms, so your level of success will be in employing several optimization tricks in an attempt to cover all your bases. Optimize your web pages BEFORE submitting your URL to search engines. Keep in mind that human-compiled web directories like Yahoo! rely on your submitted site description and keywords for proper category placement, so changing your web pages will not increase your ranking in those web directories. Modifying your web pages will only affect your ranking in "crawler" sites like Google that visit and index web sites.

Search engines used to look for specific META tags in your HTML code to help categorize and index your web pages. Over the years, many of the major search engines have stopped looking at META tags since so many web marketers have abused the technology by inserting hundreds of unrelated keywords in an attempt to trick these systems

into granting higher search rankings. While only a handful of search engines still look at META tags, you should still include them anyway since many experts believe that the presence of these extra keywords can still influence your overall placement.

```
<HEAD>
<TITLE>CodeQuiver: Time-Saving Code Snippets Organizer for
Software Developers and Web Designers</TITLE>
<META NAME="description" CONTENT="CodeQuiver is the time-
saving code snippets organizer for software developers and
web designers.">
<META NAME="keywords" CONTENT="code, snippets, syntax
coloring, source code, Mac, programming, development,
software, library, organizer">
</HEAD>
```

The Description META tag and the Keywords META tag should both be placed within the HEAD tags of your web pages. An effective description is one that avoids hyped marketing jargon and is usually limited to 20-30 words, no longer than 200-250 characters. Don't go overboard with too many keywords (try 40 or less) and don't repeat keywords. Many search engines reject site submissions that hide hundreds of repeated keywords in META tags, invisible comment tags, image ALT tags, etc.

So if META tags are not as effective as they once were, what are the primary elements that search engines currently do analyze? Here's a quick list of several site optimizations that should help improve your status with the major search engines.

TITLE Tag. This is often the phrase that is used as the hyperlink text when your site appears in search results, so including the right keywords will not only improve your ranking, but also entice consumers to click on the link.

Text Placement. Search engines will scan the top half of your web pages for relevant keyword patterns in your text headlines and paragraphs. Unfortunately, lots of HTML table tags or JavaScript code that push your text paragraphs further down the page can negatively affect your search rankings. Search engines cannot read text embedded in Flash or images.

HTML Hyperlinks. "Crawler" sites like Google index your site by following HTML links from your home page. The more links that interconnect all of your web pages, the more indexed pages that can appear listed in search results. Search engines cannot follow links embedded in Flash, imagemaps or dynamic JavaScript code.

Avoid Frames. Most search engines have trouble following links to frame pages. Since the home page is usually the main frameset page with little to no text, search rankings would surely plummet when frames are involved.

MOVING FORWARD

While there are certainly other creative avenues for driving traffic to your web site, the methods illustrated in this article should give you a solid head start in promoting your software products online. In future installments of this column, we'll explore other web traffic building techniques such as online advertising, link exchanges and revenue-sharing affiliate programs.

By Chris Kilbourn

Pursuing the Dream

From Dream to Reality

THE DREAM

Chances are, if you are reading this magazine, you are likely a coder, a hardware engineer, a systems administrator, a network manager or some mixture of all four.

But deep inside of you have a dream, a dream of being an entrepreneur.

You have code, a hardware product, or a web service you're working on that will rock the world and make you a mint — if you only knew where to start. You learned how to sling code, balance capacitors and plumb networks, not read balance sheets or write business plans, after all!

Maybe your desire to strike out on your own and be your own boss is a burning flame, or maybe an ember just sparked. Maybe you've been nurturing your idea for years and have been slowly working towards realizing the dream, or maybe it just occurred to you in a flash of insight this morning while in the shower. But you know that at some point in your life you will strike out on your own.

I've been there.

In the Spring of 1994, I found myself working at a job where my fortunes shifted from a position that I thought embraced technological innovation, professional advancement, and learning into a bleak future of technological stagnation and mind-numbing sameness enmeshed within a static bureaucracy.

Hired to perform systems analysis for a multi-unit division at a large educational institution, I poured three months of my life into assembling a plan to integrate seven different departmental information systems that included scraps of paper, HyperCard stacks, FileMaker databases, Excel spreadsheets and VAX terminal systems.

The final plan incorporated client-server technology, the use of Newton PDA's for data-collection, and would have cut the time my division's departmental managers spent poring over budget reconciliation printouts to near zero. This was cutting-edge stuff in the early 90's.

After presenting plans to my two managers, I was duly informed that there was no budget to actually implement the

plan. It seems that there were two vital bits of information they neglected to tell me during my interviews: first, there never really was any budget to implement a plan, but only budget for my position; and second, they thought it would take anyone they hired at least a year or two to pull together a plan. In the meantime, they could secure budget for implementation.

Needless to say, I was not amused.

Angry, confused, upset, and feeling misled, I retreated to my office for the next few weeks and stewed about the situation. Meanwhile, this thing called the World Wide Web was starting to pick up steam and I threw a copy of MacHTTPD on a Mac II in the back room, taught myself HTML in a day and set up a personal web site.

Working on the web server and posting copiously to USENET occupied most of my working days as I had pretty much completed the job task I was hired to do given the budget situation. It wasn't until I stumbled across a web-enabled database that the light bulb went on in my head on how to do the job I was hired to do using even more cutting-edge technology; for free, even!

I spent the next few weeks coding a demonstration for my managers, loading in departmental sample data, and re-writing the plan with these new web tools in mind. My presentation scheduled, the appointed day and hour arrived and I began my demo.

Explaining that this was brand-new, evolving technology with some current feature limitations that would shortly be addressed, I walked them through how our various departments could use the web for data collection and dissemination. As I wrapped up the presentation I laid before them the denouement: it was all free and would not require any budget. Silence settled into the room save for the gentle whirring of the HVAC and computer fans.

I will never forget the last five minutes of that meeting for as long as I live.

The managers looked at each other, looked at me, then shifted in their chairs and looked at the computer screen where I had done my demo, whereupon the division manager turned to me and asked:

"Where did you get the equipment to do this demonstration with?"

Chris Kilbourn is an independent small business, network and web infrastructure consultant. Chris is also the founder of digitalforest, Inc., <http://www.forest.net>, which offers database, application and web hosting services in addition to server colocation. When he's not out running marathons, you may contact him at chris@forest.net.

Not, "How does this work?" or "Is it really free?" or "I don't understand it, but it looks intriguing."

I was puzzled and did not know how to answer the question. I was expecting all manner of questions or comments from deployment times to resource issues, but this one came out of left field.

Confused, I explained the computer was one of our spare desktops that we kept on hand in case someone's computer went on the fritz and needed an immediate replacement.

I was then subjected to a harangue about the misuse and misguided deployment of division assets and how my time could be better spent on other tasks.

During this managerial tirade, I had a moment of personal clarity.

Namely, that it was time for me to move on and pursue a childhood dream of owning my own business.

I had greater aspirations than battling budgets and shortsighted managers for the rest of my career. The pension in the far future was enough of a reward for my co-workers daily toils, but for me, it was not.

As a network and systems administrator, I knew my way around file systems and routers well enough, but I felt lost in the wilderness when it came to figuring out how to take my idea and turn it into a business that people would actually pay me money to do. Never mind not knowing where I was going to get the money to start the business, nor understand how to market my service!

THE PLUNGE

I think that there are a lot of you out there today who are where I was almost a decade ago: sitting on a business idea, but needing some encouragement and direction in order to leap into the world of the self-employed.

Now, almost ten years after I took the risk to go into business, the company I founded, digital.forest, is still going strong. Living proof that even if you are more comfortable, as I was, in front of a LCD screen, test bench, or patch panel than in front of a group of bankers, you can still succeed in business.

MacTech also thinks that there are a lot of you out there ready to chase the American dream, and over the next year we will be publishing a series of articles written by yours truly about how to start and run a business.

I'll share practical advice about how to start and run a small technology business. There are hundreds of books out there about how to write a business plan, psyche yourself up for the entrepreneurial challenge, where to find customers, and how to raise money to get started. I read quite a few of these books as initial research before I launched digital.forest.

It wasn't until years later that I learned that those books gave short shrift to the reality that few people will really read your business plan, that starting a business is a lot harder than it appears in the movies, that there are customers you do not want, and that raising money to get the business off of the ground and keep it going can consume huge portions of your time.

I'm hoping that by sharing some of my successes and failures, along with insight and advice about the nuts and bolts of running a business, that you can only receive from someone who's been there before, you will be able to get a leg up in getting your venture off the ground, while avoiding some of the pitfalls that I fell into.

WHAT DO YOU THINK?

Space will preclude me from covering every potential topic or issue that you might want to know about, so I would like to solicit your feedback and questions in shaping future columns. Maybe you have a question about patents and trademarks, or how to value a company for sale. Or maybe you want to know how to find a good office broker, or how to evaluate an advertising agency.

Send me your questions, and I'll do my best Dear Abby impression and devote some space in each column to answering them. If there is a question that is outside of my scope of experience, I'll use my network of colleagues, accountants, bankers, lawyers, etc. to find an answer for you. If I receive enough questions about a certain topic, I'll devote a full column to it.

WHAT'S NEXT?

Over the next several months we will examine:

- **Why you would want to start a business.** The long hours, the financial risk, the glamor of cleaning out the company refrigerator; who wouldn't want to start their own business?

From frustration with a current job to realizing a childhood dream, everyone has a reason why they want to start a business. We'll take a closer look at how your motivation for starting a business can shape how you do business and define success along with exploring some of the hazards of choosing this life altering course.

- **Business planning.** Post-.com, it is now crystal clear to everyone that businesses without clear plans are doomed to failure (at least, I hope it is). Even though, in retrospect, my first business plan, (and even some subsequent ones) were laughable, they did provide a direction to move in when things were unclear.

And no, your business plan is not the cocktail napkin you sketched out your idea on!

- **Banking, legal and accounting issues.** I know that most of you would much rather be cleaning out the bit bucket, or taking a spin at front-line technical support than spend your time dealing with bankers, lawyers, and accountants. But, like regular visits to your dentist and doctor, these folks are vital to the health of your business, and you need to know how they can help you.

Believe it or not, I've met professionals in these areas that knew more than I did in some technical areas, so set some of your preconceptions aside. I'll help you ferret out the smart ones who won't gouge you, and cover the basic things you will need from them.

- **Bookkeeping and financial management.** It's all about money. You have to pay attention to it, or you won't have any of it.

Just like debugging code, an oscilloscope or a packet analyzer, having a firm grasp on business financial metrics will provide you with a powerful tool for decision-making, and assist you in evaluating business performance. We'll tour the balance sheet, income statement, and chart of accounts with an eye towards the vital bits. My mantra has always been cash flow, cash flow, cash flow, and I'll explain why.

- **Business structures.** LLC, C-corp, sole proprietor, partnership, LLP, S-corp. Learn how your business structure affects your future growth, and how to select the correct one for your type of business.

Organizing with the correct business structure has huge implications for future growth and taxes. Select the wrong one, and you could be making an expensive mistake.

- **Sales.** Arguably the most difficult thing for a technical person to learn how to do effectively. However, without selling, you'll be out of business faster than curry through a baby. Learn how to take your expert knowledge and turn it into a sales tool.

- **Marketing and advertising.** Besides taking out a two-page, 4-color spread in MacTech, there are other ways to promote your business.

From guerilla marketing to free and low-cost advertising, there are a myriad of ways to let the world know that you are in business, and worthy of having your customers send you their cash.

- **Staffing and recruiting.** With the current economic climate, looking for staff is an invitation to be bombarded by electronic and paper resumes and phone calls from potential employees and staffing firms.

Many businesses founder due to poor staffing choices and the founder's inability to let other people do work. Unless you're working as a lone ranger, you will need to learn how to hire, manage, and most importantly, delegate.

Long Distance

3.9¢ Per Minute!

Straight 6 second billing increments

Excellent rates on intrastate, intralata/toll calls and international calling with no term contract.

Toll Free (800/888/877/866) service, same low per minute rate for incoming calls.

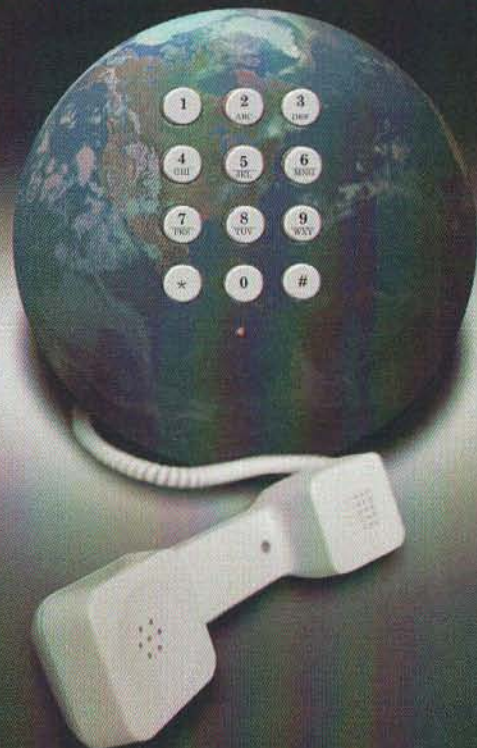
10 cents per minute calling card.

Detailed billing directly from Capsule Communications, a Covista Company.

Quality electronic and telephone customer support.

No monthly billing fee if you sign up for AUTOPAY billing option or if your bill is over \$20.00 each month.

(NOTE: \$1.00 billing fee is charged when your bill is under \$20.00 for all non-Autopay customers.)



www.lowcostdialing.com

- **Fundraising.** A black box to most, finding the money to start and grow the business can be the hardest nut to crack. Many businesses fail because they don't have enough money at the outset to grow to profitability. Business books euphemistically call this "under-capitalization."

Only call a potential investor on the second day of a stock market upswing. Bankers will tell you what they need from you in order to loan you money, you just have to ask them. Selling your product or service is the most reliable way to raise money. Friends and family want more than their money back. I'll cover these, and other secrets of raising money to get off the ground and fund growth.

- **Growth and management issues.** Growth can kill and has killed companies in the past. (Apple Computer is the exception, not the rule.) Learn how to manage your company's growth before it strangles your business.
- **Success and failure.** So, you made millions, or left nothing but a smoking hole in the ground. Now what do you do with the rest of your life?

THE FEAR OF STARTING A BUSINESS

I know that some of you are sitting out there right now, thinking about taking the plunge of becoming self-employed but you are been held back by the fear of taking the risk. I know that I was partially terrified before I made my decision to quit my job and start digital.forest. Even though my job was terrible, it did have great benefits, and the paycheck did clear when I deposited it. Letting go of that safety net is an act of faith in yourself, but it does not require you to let go of your fear.

I was afraid that I did not have what it would take to make the business a success. I was afraid of all of the debt I would incur. I was afraid that it would take time away from other things in my life that were important to me. I was afraid that there were people out there who were smarter than me, who were better-funded, who had run a business before, and who would crush my business.

What I found was that no matter my competitor's experience, stockbrokers have it right: past success does not guarantee future performance. The flip side is also true: past failures (or lack of experience) do not guarantee future failures.

I found that the debt, while painful, did not kill me. I found that it did take me away from some important things in my life, but opened up new interests and introduced me to new friends. I found that even though there were people who were smarter, better-funded and had run a business before, they were so busy running their own business, they didn't have time to pay attention to me (Some of them even helped me out from time to time).

Unless you have the emotions of a rock, you will find that fear can be the greatest enemy of your success. I've met people

who seemed to be the most self-assured and successful people in the world and they have confided in me their fear of being overextended, of their competition, and of their ability to successfully navigate their business through treacherous waters. It just comes with the territory.

You need to be aware that fear creates one of two responses in animals, human beings included: fight or flight. Fear can crush you, or it can spur you on to greatness. You need to believe in yourself and your dream and if you do not, you are not ready to pursue the dream.

Each time I found myself in a hard spot, I thought back to that moment when I realized it was time for me to start a business. The fear of returning to that type of work environment was always greater than what I faced.

Find the touchstone of your dream and use that as the bedrock upon which to stand and face your fears.

DISCLOSURE

In the interest of full disclosure, I am neither a lawyer, an accountant, nor a banker; I don't even play one on television. I'm just a networking/systems administrator who learned business management skills the hard way, by doing.

That being said, before you go off and make any sort of decision that has legal, tax, or financial implications, consult with a professional in the field.

THE REALITY

I'm hoping that these columns will help you out on the road to starting your own business or help you with your existing business. Everyone should be aware of the fact that starting a business is hard work and that most businesses will fail within their first few years. The statistics do not lie. According to the Small Business Administration, 50% of small businesses fail in their first year and 95% fail within five years.

Your goal is to be in the 5% of businesses that succeed.

Starting a business is not for the faint of heart. It will be hard work, it will take you away from other things in your life like family, friends, and hobbies, and it will require sacrifice.

If I had known how hard it would be to start digital.forest, I never would have done it. It negatively impacted my health, my relationships and my financial situation for several years. Yet now I cannot imagine not being involved in some sort of business endeavor as an owner or principal. For the rest of my life, I will always be involved in starting businesses.

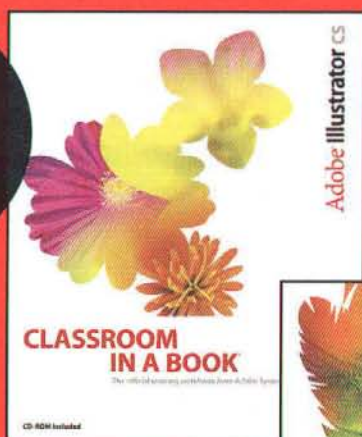
In the end, no matter your motivation for starting the business, you do it for yourself. Just be aware that the deck is stacked against you and that you need to hold your dream close to your heart in order to be successful.

Good luck!

Next month: How the decision to start a business will change your life.

design. inspire. deliver.

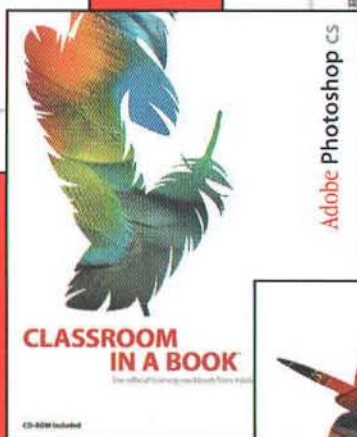
**30%
OFF**
Dec. 28, 2003–
Jan. 25, 2004!



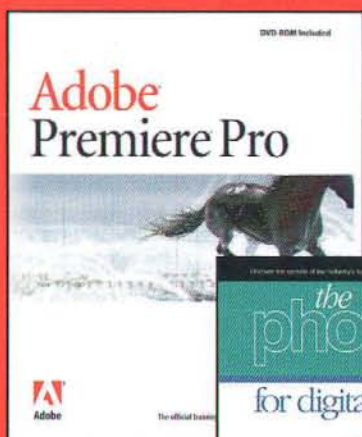
ISBN 0-321-19380-6



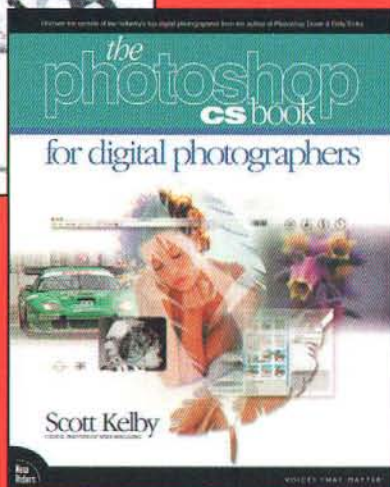
ISBN 0-201-79537-X



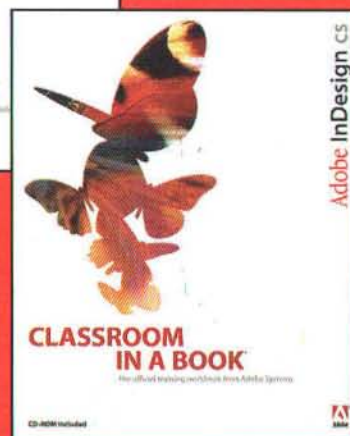
ISBN 0-321-19375-X



ISBN 0-321-19378-4



ISBN 0-7357-1411-8



ISBN 0-321-19377-6

BORDERS®
BOOKS MUSIC MOVIES CAFE



Pearlman Press

Adobe Press

New
Riders

List of Advertisers

Aladdin Knowledge Systems, Inc.	11
Aladdin Systems, Inc.	21
Big Nerd Ranch, Inc.	43
Brad Sniderman	53
DevDepot	28-29
DevDepot	55
Electric Butterfly	61
FairCom Corporation	1
Fetch Softworks	45
Full Spectrum Software, Inc.	19
IDG World Expo Corporation	50-51
Lemke Software GmbH	61
Lingo Systems	13
MacDirectory	47
Mathemaesthetics, Inc.	15
MCF Software	31
MYOB US, Inc.	17
Netopia, Inc.	IFC
Paradigma Software	23
Peachpit Press	79
Pearson Education Communications	59
piDog Software	63
PowerGlot Software	39
PR Newswire	65
PrimeBase (SNAP Innovation)	35
Prosoft Engineering, Inc.	33
Runtime Revolution Limited	BC
Seapine Software, Inc.	25
Small Dog Electronics	37
Sonnet	69
Sophos, Inc.	9
Sybase, Inc.	2-3
TechSmith Corporation	7
The Iconfactory	49
ThinkFree Corporation	41
Trango Broadband Wireless	73
Utilities4Less.com	67
VVI	57
WIBU-SYSTEMS AG	IBC

List of Products

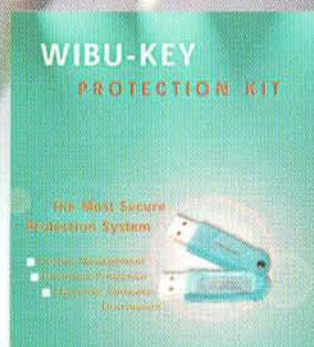
Adobe Press • Peachpit Press	79
Big Nerd Ranch • Big Nerd Ranch, Inc.	43
c-tree Plus • FairCom Corporation	1
Developers Library • Pearson Education Communications	59
Development & Testing • Full Spectrum Software, Inc.	19
Disk Utilities • Prosoft Engineering, Inc.	33
Ensharpen • TechSmith Corporation	7
Enterprise Software • MCF Software	31
Fetch • Fetch Softworks	45
Graphic Converter • Lemke Software GmbH	61
HelpLogic • Electric Butterfly	61
InstallerMaker, StuffIt • Aladdin Systems, Inc.	21
Law Offices • Brad Sniderman	53
Long Distance Phone Service • Utilities4Less.com	67
MacDirectory • MacDirectory	47
Macworld Conference & Expo • IDG World Expo Corporation	50-51
Maximizing Your Mac! • DevDepot	28-29
New Product • MYOB US, Inc.	17
piDog Utilities • piDog Software	63
PowerGlot • PowerGlot Software	39
PR Newswire • PR Newswire	65
PrimeBase • PrimeBase (SNAP Innovation)	35
Processor Upgrade Cards • Sonnet	79
Resorcerer • Mathemaesthetics, Inc.	15
Runtime Revolution • Runtime Revolution Limited	BC
Security • Aladdin Knowledge Systems, Inc.	11
SmallDog.com • Small Dog Electronics	37
Software Protection • WIBU-SYSTEMS AG	IBC
Sophos Anti-Virus • Sophos, Inc.	9
Stock Icons • The Iconfactory	49
Sybase • Sybase, Inc.	2-3
TestTrack Pro • Seapine Software, Inc.	25
ThinkFree • ThinkFree Corporation	41
Timbuktu • netOctopus • Netopia, Inc.	IFC
Translation & Localization • Lingo Systems	13
Upgrade Your Mac • DevDepot	55
Valentina • Paradigma Software	23
Visual-Report Tool Developer • VVI	57
Wireless Networking • Trango Broadband Wireless	73

The index on this page is provided as a service to our readers. The publisher does not assume any liability for errors or omissions.

WIBU-KEY Software Protection

Be sure to order a **WIBU-KEY Protection Kit** and find out why small software companies to Fortune 500 enterprises are switching to the WIBU-KEY Security, not Obscurity model for all of their software licensing needs, including:

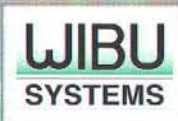
- Common API across all platforms and hardware
- Hardware-based code and data encryption
- Field-upgradable and reusable hardware
- The most cost-effective network licensing solution available
- The only system to employ RID/RED and AXAN security
- Engineered and manufactured to exacting ISO9001 standards



Test the
WIBU-KEY
Protection Kit
sales@griftech.com

Call 800-986-6578

The Key is in Your Hands!



WIBU-SYSTEMS USA, Inc.
Seattle, WA 98101
Email: info@wibu.com

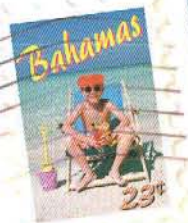
www.griftech.com
www.wibu.com

Protection Kits also available at:

Belgium wibu@impakt.be, Denmark lean@danbit.dk, Finland finbyte@finbyte.com, France info@neol.fr, Hungary info@mrsoft.hu, Japan info@suncarla.co.jp, Jordan, Lebanon starsoft@cyberia.net.lb, Korea dhkimm@wibu.co.kr, Luxembourg wibu@impakt.be, Netherlands wibu@impakt.be, Portugal dubit@dubit.pt, Thailand preecha@dpf.co.th, United Kingdom info@codework.com, USA sales@griftech.com

It'll Give You A Life.

Nassau Beach, Bahamas - Sun, sand and soft breezes make this one of the most relaxing vacation spots in the world. Don't forget the cool, fruity drink!



Dear Revolution 2.0,

Thanks for the
great vacation!

Missing you terribly,

Love,

Cecil

Revolution 2.0
c/o My Computer 64.23.0.192
24-7 Relief From Aggravation
(formerly DullAndDreary Coders)
The Corporate World, #1-EASY

But A Geek Is Always A Geek.

**Revolution 2.0: the English like language designed around the way you think.
Develop and deliver on Mac OS X, Windows and Linux
(not to mention 10 other platforms).**

Take the English language, add elegant XML, more SQL databases than you care to learn, intrinsic video capture, Unicode, CGI scripting, sophisticated Reports, and build your solution faster than anyone else. Jaguar, Panther, XP? Revolution lets you eat platforms for breakfast. And speaking of eating, our Cookbook of examples gets you up, running, and productive NOW. You can join the Revolution for as little as \$99...

Build with it, deliver with it, love it - and still have time for vacations.

And for a limited time, your MacTech special lets you get 100% of the Revolution for 15% off - go to special.runrev.com NOW. Thousands of developers have already joined.

Don't let the Revolution in modern coding start without you...

Software At The Speed Of Thought



Software Development & Consultancy

Runtime Revolution • 91 Hanover Street • Edinburgh EH2 1DJ • UK
Phone +44 (0)131 718 4333 • Fax +44 (0)131 718 4334 • www.runrev.com • Email info@runrev.com